Higher-order structure in networks: Construction and its impact on dynamics

Martin Ritchie

A thesis submitted for the degree of Doctor of Philosophy

University of Sussex
Department of Mathematics

April 2016

Declaration

I hereby declare that this thesis has not been, and will not be, submitted in whole or in part to another University for any other academic award. Except where indicated by specific stated in the text, this thesis was composed by myself and the work contained therein in my own.

signature .....

Martin Ritchie

University of Sussex

MARTIN RITCHIE

THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

HIGHER-ORDER STRUCTURE IN NETWORKS: CONSTRUCTION AND IMPACT ON DYNAMICS

SUMMARY

Networks are often characterised in terms of their degree distribution and global clustering coefficient. It is assumed that these provide a sufficient parametrisation of networks. However, since the global clustering coefficient is only sensitive to the total number of triangles found in the network, it is evident that two networks could have the same number of triangles but significantly different higher-order structure, i.e., the topologies that result from the placement of closed subgraphs around nodes. The two main objectives of my work are: (1) developing network generating algorithms and network-based epidemic models with controllable higher-order structure and (2) investigating the impact of higher-order structure on dynamics on networks.

This thesis is based on three papers, corresonding to Chapters. 3, 4 and 5. Chapter 3 presents a novel higher-order structure based network generating algorithm and subgraph counting algorithm. Chapter. 4, generalises a previously proposed ODE model that accurately captures the time evolution of the susceptible-infected-recovered (SIR) dynamics on networks constructed using arbitrary subgraphs. Chapter. 5, improves, extends and generalises the network generating algorithms proposed in the previous two papers. All three chapters demonstrate that for a fixed degree distribution and global clustering, diverse higher-order structure is still possible and that this structure will impact significantly on dynamics unfolding on networks. Hence, we suggest that higher-order structure should receive more attention when analysing network-based systems and dynamics.

Acknowledgements

Istvan and Luc, you have both taught me so much and so well. Knowing what I know now, starting over again and given any choice of supervisors I would pick you both again.

I am eternally grateful to the *Engineering and Physical Sciences Research Council* (EPSRC) and the *University of Sussex* for fully funding my PhD. And, again my supervisors for securing my funding. This has been a life changing opportunity.

I would also like to thank Thomas House whom I was lucky enough to collaborate with on my first paper.

I am truly fortunate to have been blessed with an incredible family: Mum, Dad, the cows (my sisters for the uninitiated), Ben and John. I worry immensely about writing a list of friends (which would run into the hundreds, literally) and accidentally leaving some one off, getting the order wrong or otherwise cocking it up in some other spectacular way. You know that I love you all, immeasurably.

List of publications and author contributions

- **Higher-order structure and epidemic dynamics in clustered networks.**
  M. Ritchie, L. Berthouze, T. House, and I. Z. Kiss.
  Journal of Theoretical Biology, 348:21 – 32, 2014.

  - M. Ritchie conceived the overall goals of the study and the analysis, developed and implemented the bespoke *clustered configuration model* network generating algorithm, provided the majority of the calculations, generated the majority of the data, data visualisation, wrote the first draft and subsequent revisions.
  - L. Berthouze conceived the overall goals of the study and the analysis, developed and implemented the *motif counting algorithm*, provided data using the motif counting algorithm, contributed to writing/revising the paper, and closely supervised the work of M. Ritchie
  - T. House developed and implemented the *motif decomposition algorithm* and associated theoretical calculations and contributed to writing/revising the paper.
  - I.Z. Kiss conceived the overall goals of the study and the analysis, contributed to writing/revising the paper and closely supervised the work of M. Ritchie

- **Beyond clustering: mean-field dynamics on networks with arbitrary subgraph composition.**
  M. Ritchie, L. Berthouze, and I.Z. Kiss.
  Journal of Mathematical Biology, 72(1-2):255–281, 2016.

  - M. Ritchie conceived the overall goals of the study and the analysis, developed and implemented the *hyperstub configuration model* associated network generating algorithm and code generating algorithm, provided all theoretical calculations, generated the majority of the data, provided all data visualisations, wrote the first draft and subsequent revisions of the manuscript.

  - L. Berthouze conceived the overall goals of the study and the analysis, contributed to writing/revising the paper, provided data using the *motif counting algorithm* and closely supervised the work of M. Ritchie.
  - I.Z. Kiss conceived the overall goals of the study and the analysis, contributed to writing/revising the paper and closely supervised the work of M. Ritchie.

- **Generation and analysis of networks with a prescribed degree sequence and subgraph family: Higher-order structure matters.**
  M. Ritchie, L. Berthouze, and I. Z. Kiss.
  Journal of Complex Networks (in press, 2016).
  Also available at: http://arxiv.org/abs/1512.01435

  - M. Ritchie conceived the overall goals of the study and the analysis, developed and implemented the *underdetermined sampling algorithm* and *cardinality matching algorithm*, provided the majority of the computational evidence, theoretical calculations, provided all data visualisations, wrote the first draft and subsequent revisions of the manuscript.
  - L. Berthouze conceived the overall goals of the study and the analysis, provided data using both the *motif counting algorithm* and *complex contagion* dynamics, contributed to writing/revising the paper and closely supervised the work of M. Ritchie.
  - I.Z. Kiss conceived the overall goals of the study and the analysis, developed and implemented the *underdetermined sampling algorithm*, contributed to writing/revising the paper and closely supervised the work of M. Ritchie.

Contents

Chapter 1

Motivation and thesis overview

1.1 Motivation

Networks represent elements of a system by nodes and interactions between elements as edges. Figure 1.1 shows a small portion of the internet, a physical network where servers and routers form the nodes and the copper wires or fibre optic cables that connect them form the edges. The internet hosts the world wide web, itself a network but digitally abstracted. In this case web pages form nodes and hyper-links are the edges that connect nodes. When a network is used to represent a population people are represented by nodes and their contact between individuals is represented with edges. Nodes may also represent more than one type of element of a system. Figure 1.2 shows a network of different languages spoken in a small community. This type of network is known as a *bipartite graph* where the nodes form two disjoint sets: people and languages.

Aside from their eloquent visualisation of complex systems networks are highly adept at quantitatively and qualitatively characterising the structure of complex environments. The importance of this cannot be understated since a network's structure will significantly affect its *function* [72]. It is well known that degree heterogeneity has a major impact on the epidemic threshold [51], and that highly connected nodes can be preferentially targeted in order to limit spread [2, 12]. Networks that are assortatively mixed by degree will significantly reduce the final size of an epidemic compared to dis-

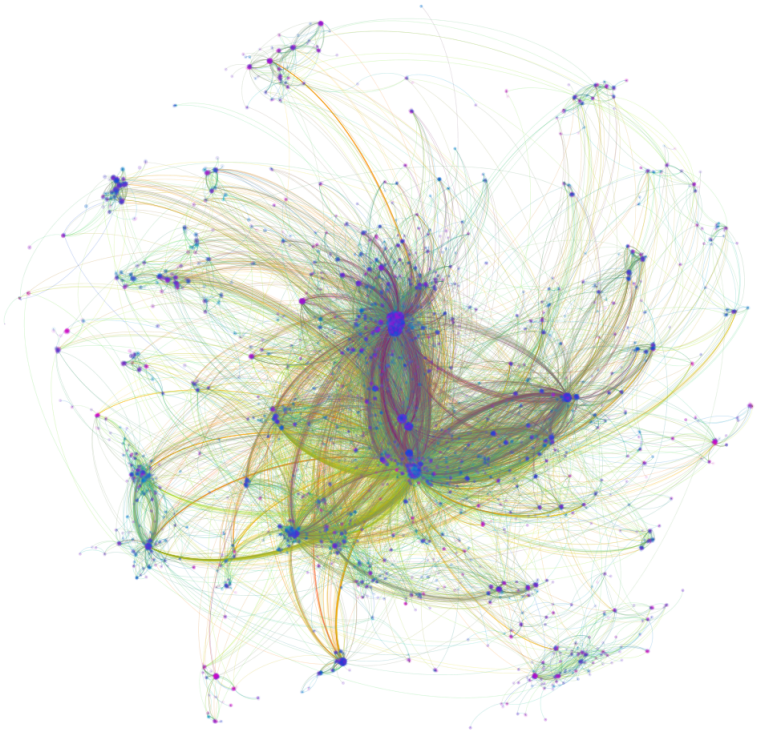


Figure 1.1: A small fraction of the internet, data taken from the Route Views project, see [1]. This figure, and all other network visualisations in the thesis, have been produced using *Gephi*, see [8]. The nodes are coloured so that lower- and higher-degree nodes appear towards the red and violet ends of the colour spectrum respectively. In this figure there are: $N = 22,963$ nodes, on average each node is incident to 4.2 edges and the maximum degree is 2390

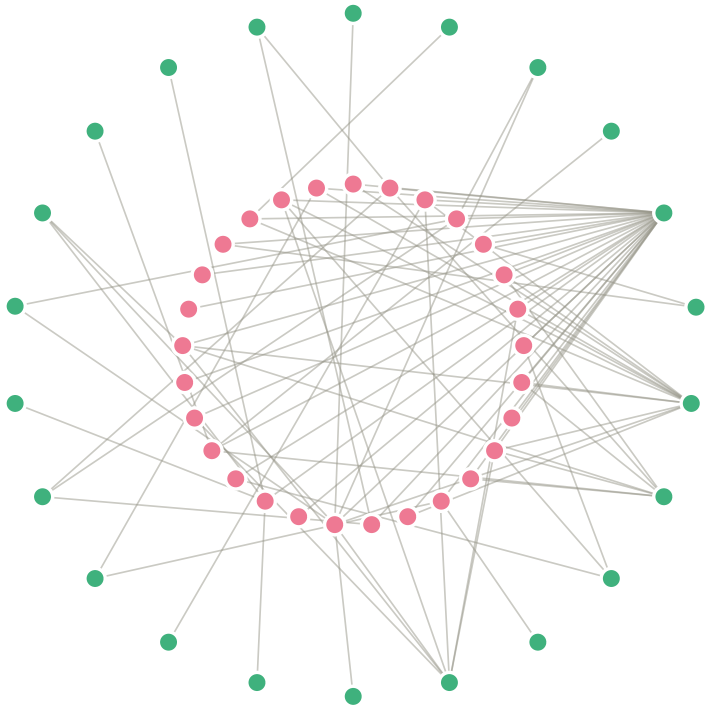


Figure 1.2: The network of languages spoken in the department of Mathematics at the University of Sussex. Green and pink nodes represent languages and people respectively. English forms the most central node. Similarly, French and Spanish form the second and third most central nodes in the network.

assortative networks [52]. Clustering will also inhibit the spread of an epidemic relative to non-clustered networks [20]. All of these aforementioned properties also impact on the epidemic threshold defined loosely as the number of new infections produced by a typical infectious individual introduced in a fully susceptible population [33, 52]. In brain networks, the growth of the network and dynamics are intimately coupled [11]. In some cases the dynamics that the network host are a direct result of network architecture [19, 59]. Causal links have been found between network structure and mental illness with great implications for the individual concerned [42].

The most basic structural description is *edge/link density*, the ratio of the number of edges to all potential edges. However, between having no connections or all possible connections there is an extremely large number of possibilities, see Figure 1.3. A network's *degree distribution*, a discrete probability distribution that describes the probability p_k of finding a node incident to k edges, better describes this spectrum. The degree distribution is probably the most important descriptor of a network and plays a major role in determining the behaviour of dynamics that run on networks.

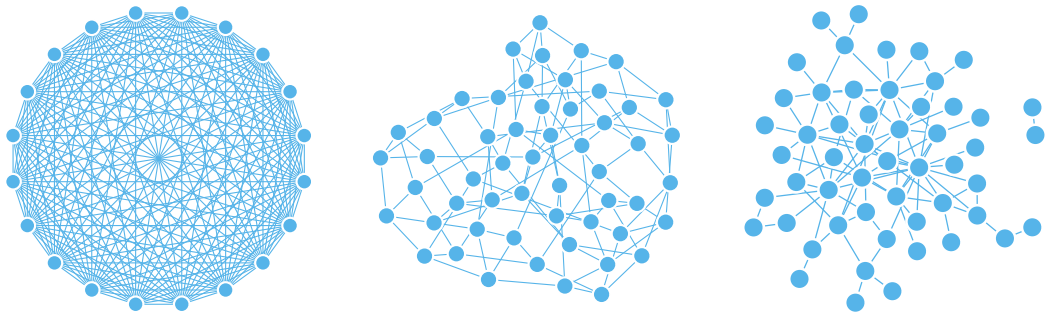


Figure 1.3: The complete network, left, has every possible connection realised, this network has unit density, see Definition. 3. Each node in the homogeneous network is connected to exactly four other nodes and the heterogeneous networks have an average degree of four, that are relatively sparse in comparison. As can be seen from the homogeneous and heterogeneous networks with equal density, density alone is a poor determinant of network structure.

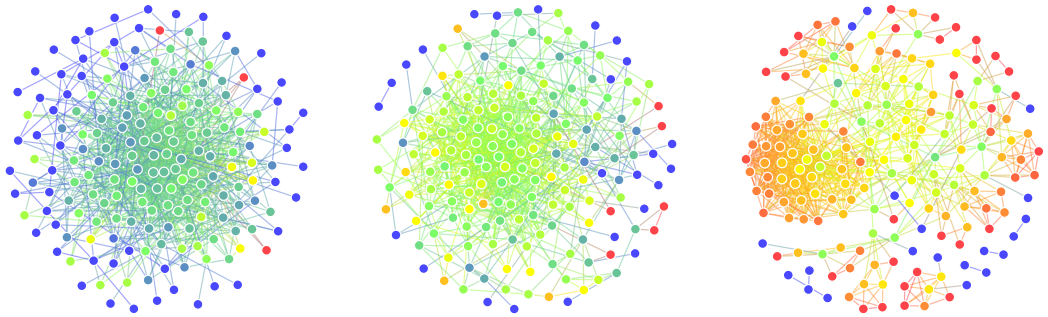
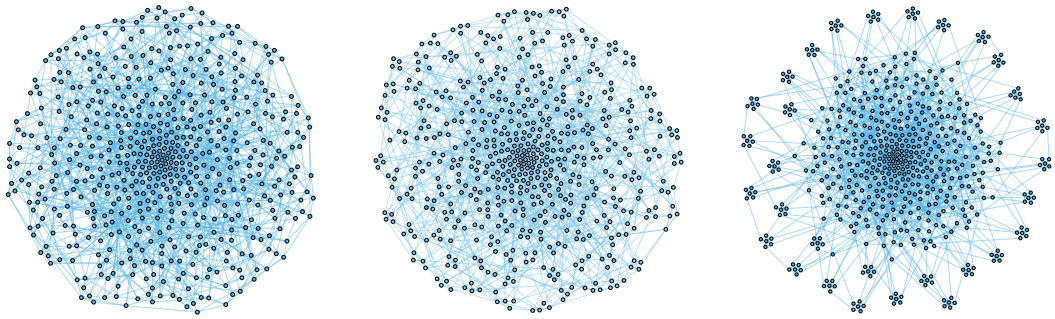


Figure 1.4: These three networks share the same degree distribution but exhibit increasing levels of clustering. The network has been coloured so that nodes incident to few triangles are coloured towards the blue end of the spectrum and nodes incident to many triangles are nuanced towards the red end of the spectrum. The geometrical embedding of these networks has been selected only for aesthetic purposes.

Figure 1.4 shows three networks that share the same degree distribution but exhibit significant structural differences. This has been achieved by increasing the networks' *global clustering coefficient*, the probability of finding two connected nodes that share a common neighbour. By increasing a network's global clustering coefficient, nodes are obliged to connect to other nodes that already reside within their close neighbourhood. This effect has been commonly observed in real world networks, see [11, 56, 72, 78], and is known to significantly impact dynamics that run on the network [20, 55]. Therefore when making comparisons between networks it is essential that one correctly accounts for both their degree distributions and global clustering coefficient. But how accurately does the degree distribution *and* global clustering coefficient alone describe network structure?

Figure 1.5 shows three networks that share the same degree distribution and global clustering coefficient. This figure demonstrates how the degree distribution and clustering alone may not sufficiently constrain network structure. Obviously, this observation has already been made in previous studies, see [6, 20, 24], for example. Clustering yields only information about the total number of triangles in a network. Consequently, this



$C = 0.202.$ $C = 0.203.$ $C = 0.206.$

Figure 1.5: Each of the above networks has been constructed using 600 nodes such that every node is incident to 6 edges, the networks share the same global clustering coefficient (denoted by C) but have been constructed using different subgraphs. Reading from left to right the networks are constructed with non-overlapping to densely overlapping triangles. Due to the number of triangles being constant between them the networks have become increasingly modular with relatively isolated clumps of overlapping triangles being weakly connected to the main component. The geometrical embedding of these networks has been selected only for aesthetic purposes.

measure omits two key pieces of information: (A) how triangles are distributed amongst and around nodes and (B) how triangles are arranged, e.g., sharing edges resulting in compound structures. In this thesis, we shall refer to such compounds, i.e., cycle based subgraphs of four nodes or more, as *higher-order structure*. Currently few models of network-based dynamics and network generating algorithms exists that can control higher-order structure. So, little is understood about its function or how it impacts on dynamics that run on networks. Creating such algorithms and network models, and subsequently investigating the impact of higher-order structure is the central theme of this thesis.

1.2 Thesis overview

For my PhD I wrote and published three papers that form the backbone of this thesis and are presented in chronological order in the following. Common to all three papers is the notion of using subgraphs as building blocks in a networks' construction in a configuration model type algorithm. The types of networks the algorithms are able to produce become increasingly sophisticated as the thesis progresses. Essentially, all three papers use their respective algorithms and models to investigate diversity in network structure, under the constraints of equal degree distribution and global clustering coefficient, and the impact of such diversity on network based dynamics. Before I present my published work Chapter. 2 introduces the topic more thoroughly and also contains my literature review. Please note that the, *list of publications and author contributions*, at the start of this thesis details the original contributions found in this document.

Chapter. 3, based on my first paper, presents two novel network generation algorithms and compares these to existing methods. By parametrising the ensemble of algorithms in the same way, targeting the same degree distribution and global clustering coefficient, I show that the resulting networks are quantitatively and qualitatively different. In addition to the new algorithms, my results indicate that when network structure is constrained by degree distribution and clustering alone structural diversity is still possible and that this diversity will manifest in dynamics that run on the networks. The original abstract for this paper reads as follows:

> *Clustering is typically measured by the ratio of triangles to all triples regardless of whether open or closed. Generating clustered networks, and how clustering affects dynamics on networks, is reasonably well understood for certain classes of networks [32, 76], e.g., networks composed of lines and non-overlapping triangles. In this paper we present two novel network generation algorithms: the clustered configuration model and the motif decomposition algorithm, the former being my own original contribution. Using these models, alongside existing methods, we generate networks which, despite having the same degree distribution and equal clustering, exhibit*

different higher-order structure, specifically, overlapping triangles and other order-four (a closed network motif composed of four nodes) structures. To distinguish and quantify these additional structural features, we develop a new network metric capable of measuring order-four structure which, when used alongside traditional network metrics, allows us to more accurately describe a network's topology. Then using SIS (Susceptible-Infected-Susceptible) and SIR (Susceptible-Infected-Recovered) dynamics we investigate computationally how differences in higher-order structure impact on epidemic threshold, final epidemic or prevalence levels and time evolution of epidemics. Our results suggest that characterising and measuring higher-order network structure is needed to advance our understanding of the impact of network topology on dynamics unfolding on the networks.

Chapter. 4, based on my second paper, generalises a previously proposed ordinary differential equation (ODE) model that captures the time evolution of susceptible-infected-recovered (SIR) dynamics on networks constructed using line and triangle subgraphs. The original model was capable of predicting dynamics on networks constructed using only complete subgraphs. I develop the necessary framework to include subgraphs of arbitrary connectivity. This framework is presented alongside a bespoke code generating algorithm that generates and solves the ODEs for a specific set of subgraphs. Using this model I find that SIR dynamics are sensitive to network structure beyond that specified by the degree distribution and global clustering coefficient, so called higher-order structure. The original abstract for this paper reads as follows:

Clustering is the propensity of nodes that share a common neighbour to be connected. It is ubiquitous in many networks but poses many modelling challenges. Clustering typically manifests itself by a higher than expected frequency of triangles, and this has led to the principle of constructing networks from such building blocks. This approach has been generalised to networks being constructed from a set of more exotic subgraphs. As long as these are fully

connected, it is then possible to derive mean-field models that approximate epidemic dynamics well. However, there are virtually no results for non-fully connected subgraphs. In this paper, we provide a general and automated approach to deriving a set of ordinary differential equations, or mean-field model, that describes, to a high degree of accuracy, the expected values of system-level quantities, such as the prevalence of infection. Our approach offers a previously unattainable degree of control over the arrangement of subgraphs and network characteristics such as classical node degree, variance and clustering. The combination of these features makes it possible to generate families of networks with different subgraph compositions while keeping classical network metrics constant. Using our approach, we show that higher-order structure realised either through the introduction of loops of different sizes or by generating clustered networks based on different subgraphs, leads to significant differences in epidemic dynamics despite controlling for basic network metrics.

Chapter. 5, based on my third and final paper, presents two network generation algorithms based on the clustered configuration model (presented in Chapter. 3) and the hyperstub configuration model (presented in Chapter. 4). *Improvements on Chapter. 3:* The clustered configuration model operates by selecting a random configuration of subgraphs to be allocated to nodes depending on their degree, but lacked generality in respect to choices in the degree distribution and subgraph choices. Chapter. 5 shows how to relax both of these constraints. *Improvements on Chapter. 4:*The hyperstub configuration model allows for sequences of subgraphs to be specified *a priori* but at the loss of control of the degree distribution. Chapter. 5 constructs networks using sequences of subgraphs but in such a way that the degree sequence is preserved. Using these algorithms to construct networks and traditional metrics to analyse the resulting networks I show that diversity is still possible between networks of equal degree distributions and global clustering coefficients. Furthermore, I go on show that this diversity impacts on the behaviour of susceptible-infected-secovered (SIS), SIR and complex contagion dynamics and therefore matters. The original abstract for this paper reads as follows:

Designing algorithms that generate networks with a given degree sequence while varying both subgraph composition and distribution of subgraphs around nodes is an important but challenging research problem. Current algorithms lack control of key network parameters, the ability to specify to what subgraphs a node belongs to, come at a considerable complexity cost or, critically, sample from a limited ensemble of networks. To enable controlled investigations of the impact and role of subgraphs, especially for epidemics, neuronal activity or complex contagion, it is essential that the generation process be versatile and the generated networks as diverse as possible. In this paper, we present two network generation algorithms that use subgraphs as building blocks to construct networks preserving a given degree sequence. Additionally, these algorithms provide control over clustering both at node and global level. In both cases, we show that, despite being constrained by a degree sequence and global clustering, generated networks have markedly different topologies as evidenced by both subgraph prevalence and distribution around nodes, and large-scale network structure metrics such as path length and betweenness measures. Simulations of standard epidemic and complex contagion models on those networks reveal that degree distribution and global clustering do not always accurately predict the outcome of dynamical processes taking place on them. We conclude by discussing the benefits and limitations of both methods.

Chapter. 6, the discussion, compares and consolidates my contributions and findings as well as discusses potential future work relating to my research.

Chapter 2

Introduction

2.1 Network characterisation

Networks are composed of nodes and edges. Node represent elements of a system and edges connecting nodes represent interactions between elements, for example, people and friendships, respectively. Networks can be directed, i.e., each edge is a one-way street, or undirected. This thesis considers only the latter. In this thesis networks are encoded by an *adjacency matrix*, $A \in \{0,1\}^{N\times N}$, where N denotes the number of nodes. Two nodes, i and j share an edge if $A(i,j) = A(j,i) = 1$, or more generally, $A = A^T$ (as the network is undirected). In addition to networks being undirected we also require them to be *simple*. In a simple network multiple connections between nodes and self-loops, $A(i,j) > 1$ and $A(i,i) = 1$ respectively, are not permitted.

2.1.1 Degree sequence and distribution

The number of edges incident to a node is referred to as the *degree* of a node.

Definition 1. *The* degree sequence *of a network is a sequence of natural numbers, that may include zero, which specifies how many edges originate from each node. More specifically* $D = (d_1, d_2, \ldots, d_N)$ *where* d_i *and* N *denote the* degree *of node* i *and number of nodes in the network respectively. In undirected networks the degree sequence may be obtained by adding the adjacency matrix row- or column-wise.*

The sum of the degree sequence gives the number of edges in the network doubly counted, since each edge is counted from both nodes that connect; this value is denoted by $2m = \sum_{i,j=1}^{N} A_{ij}$. The degree sequence is a single realisation of a network's *degree distribution.*

Definition 2. *The degree distribution of a network is a discrete probability distribution, p_k, which denotes the probability of finding a node incident to k edges.*

The total number of unique edges is denoted by m, and can be computed from the adjacency matrix as above or from the degree distribution, i.e., $m = \langle k \rangle N/2$, where $\langle k \rangle$ denotes the first moment of the distribution. The total number of possible edges between N nodes is $\binom{N}{2}$.

Definition 3. *The* density *of a network is the ratio of its edges to the total number of possible edges, i.e.,*

$$d^* = \frac{m}{\binom{N}{2}},$$

where $0 \leq d^* \leq 1$. A network with $d^* = 1$ is said to be complete.

In between having no edges or being complete there is considerable scope for variation, even at the same edge density, and it is therefore the degree distribution which better describes this spectrum. Necessarily this feature has been exhaustively investigated and is well understood, see [7, 9, 54, 56] for example. However, even with a fixed degree sequence there are numerous ways in which the nodes may connect.

2.1.2 Degree correlations

Another important feature of network topology is how nodes of different degrees mix. For example, do high-degree nodes preferentially connect to other high-degree nodes, so called *assortative mixing*? Or do their edges tend to terminate at low-degree nodes, *disassortative mixing*? The extent of this mixing is measured by the *assortativity coefficient* [53].

To compute the assortativity coefficient first I shall use the *excess degree distribution* to calculate degree-degree covariance. The intuition behind this distribution is the

following: select an edge at random and follow this edge to one of the nodes that it connects. The distribution that describes the degree of this node, not counting the one by which the node was selected, is the excess degree distribution. q_k, denotes the probability of finding a node with excess degree k at the end of a randomly selected edge. Using this distribution, next define the joint distribution of finding nodes with degree k and j at the end of a randomly selected edge as e_{kj} that follows the following summation rules

$$\sum_{j=1}^{m}\sum_{k=1}^{m} e_{jk} = 1, \quad \sum_{j=1}^{m} e_j = q_k, \tag{2.1}$$

where m denotes the number of edges. Note that the calculations in this subsection count over edges and not nodes. In a network where there is no covariance between nodes this distribution takes the value $q_k q_j$. Otherwise degree-degree covariance may be computed by

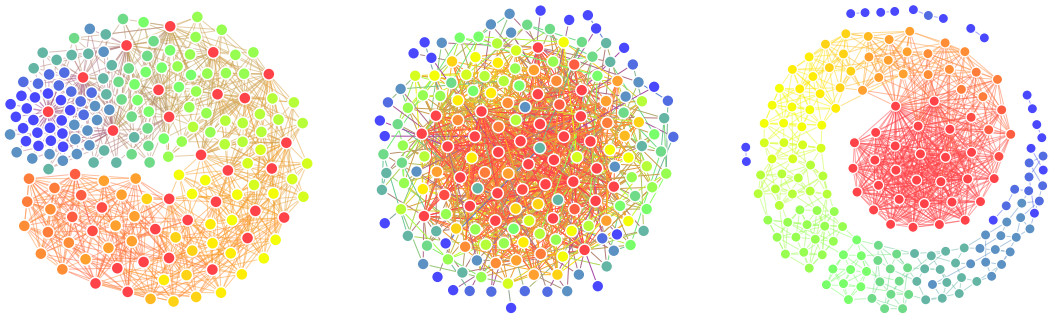
$$\langle jk \rangle - \langle j \rangle \langle k \rangle = \sum_{j=1}^{m}\sum_{k=1}^{m} jk(e_{jk} - q_j q_k) \tag{2.2}$$

This has maximum value for networks where nodes only connect to other nodes of identical degree, i.e., $e_{kj} = q_k \delta_{kj}$, where δ_{kj} denotes the Kronecker delta. This value is the variance of q_k denotes σ_q^2 and dividing Equation. (2.2) by this value yields the Pearson correlation coefficient.

Definition 4. *The* assortativity coefficient *is denoted by* $r: \ 1 \leq r \leq 1$. *Networks with values of:* $0 \leq r \leq 1$ *or* $-1 \leq r \leq 0$ *are said to be assortatively or disassortatively mixed respectively and can be computed directly from the adjacency matrix by the following*

$$r = \frac{m^{-1}\sum_{i=1}^{m} j_i k_i - \left[m^{-1}\sum_{i=1}^{m} \frac{1}{2}(j_i + k_i)\right]^2}{m^{-1}\sum_{i=1}^{m} \frac{1}{2}(j_i^2 + k_i^2) - \left[m^{-1}\sum_{i=1}^{m} \frac{1}{2}(j_i + k_i)\right]^2}.$$

Figure 2.1 shows three networks of varying assortativity coefficient. The great majority of networks that are studied in this thesis have a neutral assortativity coefficient; nevertheless, assortativity is relevant and shall be referred to in subsequent sections.



$-1 \leq r \ll 0$ $\quad$ $r = 0$ $\quad$ $0 \ll r \leq 1$

Figure 2.1: These three networks share the same degree sequence but different *assortativity coefficients*. The nodes are coloured so that low- and high-degree nodes correspond to the red and blue ends of the spectrum respectively. However, the networks on the left and right have been re-wired so that nodes of similar degree avoid one another or preferentially connect respectively.

2.1.3 Transitivity

The analysis of real world networks has revealed that many of them have a relatively large prevalence of triangles, three nodes that share every possible connection [11, 55, 56, 72, 78]. Analogous to edge density, the *global clustering coefficient* is a measure of triangle density, and is the broadest possible measure of transitivity.

Definition 5. *The* global clustering coefficient *is a measure of network transitivity. It is the probability of finding two neighbours of a given node that also share an edge forming a triangle and shall be denoted by C and can be computed directly from the adjacency matrix by the following formula [33, 77]*

$$C = \frac{trace(A^3)}{\|A^2\| - trace(A^3)},$$

where $0 \leq C \leq 1$ and $\|A^2\|$ denotes the sum of all elements of A^2.

The numerator of this expression gives the number of triangles, three nodes that share every possible connection between them. The denominator gives the number of

triangles plus the number of triples of nodes connected by two edges. A network with a global clustering coefficient of $C = 0$ will not contain any triangles. Related to the global clustering coefficient is the *local clustering coefficient*, that measures clustering around a specific node.

Definition 6. *The* local clustering coefficient*, denoted C_l, is the number of triangles incident to a node divided by the number of potential triangles incident to that node. For a node of degree k, it is given by:*

$$C_l = \frac{\triangle}{\binom{k}{2}}$$

where $\triangle$ denotes the number of uniquely counted triangles incident to the node. The local clustering coefficient is only defined for nodes with degree $k \geq 2$.

Clustering is known to improve the local efficiency of networks, in particular, it may enhance signal propagation, synchronizability and computational efficiency, which is highly advantageous to many network based dynamics [70, 72, 78]. Clustering is also present is social networks and, thus it is important to incorporate it into network-based models of spreading, be it epidemics, information or rumours. However, developing such models that can accurately capture clustering, as well as obey other constraints such as control of the degree distribution, has proven to be a challenging problem.

2.1.4 Motifs, subgraphs and higher-order structure

The edge density, assortativity coefficient and global clustering coefficient all provide macroscopic descriptions of network structure. The degree distribution and local clustering coefficient are common examples of local network properties. Where local clustering is a metric specific to only triangles, *motifs* provide a relatively general way of describing the local structure around nodes using *motifs* that they define to be as: are patterns of interconnections occurring in complex networks at numbers that are significantly higher than those in equivalent randomised, using degree-preserving rewiring, networks [49, 69]. Milo *et al.* and Shen-Orr *et al.* found that certain arrangements of nodes, such as those shown in Figure 2.2, appeared in biological and technological networks with greater frequency than what one would expect *at random* [49, 69].

To randomise the networks edges have been swapped following a degree-distribution preserving rewiring algorithm. Such a process will destroy network clustering, therefore, this definition depends on the type of rewiring procedure used to generate the null model. This work suggests that (1) the motif composition varies depending on the network function and (2) motifs may arise due to specific constraints placed on networks as they evolve, dictated by the dynamics which they host. Since clustering

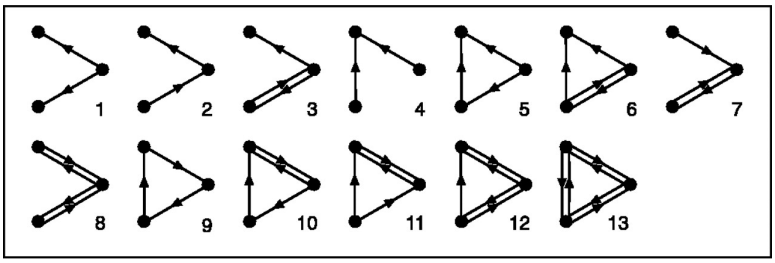


Figure 2.2: The family of three node connected subgraphs studied by Milo *et al.* in their seminal work on network motifs [49]. Note that these motifs are directed.

measures only triangles, the research community has looked at characterising networks by their motif or *subgraph* composition, and this approach has gained significant traction [11, 19, 28, 29, 62, 63].

Definition 7. *A* subgraph *is a subset of nodes and edges found in a network.*

Figure 2.3 displays a few example subgraphs that are commonly studied in this thesis.

It is important to consider that the definition of a motif will exclude subgraphs that appear with average frequency, or that are otherwise infrequent, even though they could serve an important function. Additionally, what will be defined as a motif also depends on the type of re-wiring scheme used to generate the null model. To remedy this it was suggested that networks could be characterised, and subsequently compared, using their *graphlet degree distributions* [62]. The graphlet degree distribution is a generalisation of the degree distribution which specifies the frequency with which graphlets appear around nodes. Knowing many of these distributions for a network will define much of the network structure.

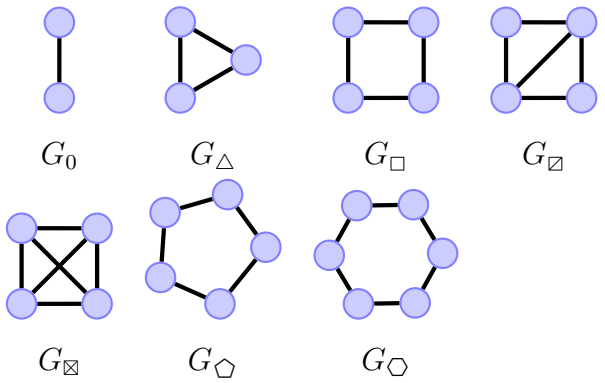


Figure 2.3: A few example subgraphs with their symbolic labels. These subgraphs are undirected and are those focused on in this work.

Definition 8. *A* graphlet *is an induced subgraph, a subset of nodes and edges that contains all edges between the nodes that are found in a network.*

Consequently, subgraphs and their distributions have started to be more commonly incorporated into network construction algorithms and predictive models of dynamics on networks [19, 24, 28, 32, 58]. But, knowing the distributions of *all* subgraphs in a networks is not plausible for networks of realistic size. Accordingly, it is more pragmatic to approximate network structure using smaller families of subgraph distributions. Where this has been investigated there is a consensus that a relatively modest number of subgraph distributions is enough to accurately reproduce many network metrics [30, 62]. However since this method is an approximation the exact topologies around nodes will still be unknown. For example, when placing many subgraphs around nodes a *natural* side-effect may be the compounding of subgraphs into more complex topologies, in addition to what is specified by the subgraph distribution.

Definition 9. Higher-order structure*: topologies that result from the placement and compounding of subgraphs around nodes that is often not captured by classical network metrics. For example, compound triangles forming hexagons.*

Higher-order structure has received very little attention to date. However, previous work has suggested that fixing degree distribution and clustering still allows to generate

distinct and diverse networks [6, 20], whilst others suggest that knowing much about the arrangement of triangles, without going beyond triples, is enough to determine much of the network structure [30]. A central premise of this work is that higher-order structure is both interesting and important, and it is our goal to present evidence to back this statement. Currently, there are few network generating algorithms and models that have the necessary features to isolate the effects of higher-order structure, and these must first be developed. In the following, we review the current arsenal of network generating algorithms and models that have provided the foundations for this work.

2.2 Network models and generation

2.2.1 Erdős-Rényi Random graphs

The most simple algorithm for the generation of simple undirected networks is that of Erdős-Rényi Random graphs [15], henceforth referred to as ER random graphs. It could be argued that this pioneering work in graph theory is the forefather of the contemporary study of random graphs, networks and *complex networks*, networks with structure more sophisticated than that found in ER random graphs.

To create a network following this method, first a network of N nodes is initiated. Then each pairing of nodes is considered in turn, and a connection is formed with probability p, or with probability $1-p$ the nodes are left unconnected. In a network of N nodes there are $\binom{N}{2}$ potential pairings of which an average of $p\binom{N}{2}$ will be realised. From this information it is possible to compute the resulting degree distribution. Each node in the network is connected to any other of the $N-1$ nodes with probability p. Therefore the probability of a node being connected to exactly k other vertices is $p^k(1-p)^{n-1-k}$. There are $\binom{N-1}{k}$ ways to choose k other nodes, so the probability of a node having degree k is

$$p_k = \binom{N-1}{k} p^k (1-p)^{N-1-k},$$

which is the probability mass function of the Binomial distribution. However, since not all networks exhibit a binomial distribution of edges, this model is limited in application.

2.2.2 The configuration model

The *configuration model* builds on the idea of ER random graphs but in this model one can fix the degree sequence *a priori*. This model was originally presented by Bollobás in [10] and later popularised by Newman in [56]. The configuration model features heavily in this work and therefore a relatively thorough review of this method is presented. To construct a configuration model network for a given degree sequence, $D = (d_1, d_2, \ldots, d_N)$, perform the following steps:

1. initialise a network of N nodes,
2. allocate each node with a number of half-edges or stubs specified by D so that node i is incident to d_i stubs ,
3. select a stub uniformly at random,
4. select a second stub uniformly at random from all remaining stubs,
5. join the two stubs to form an edge,
6. return to step 3 until all stubs are paired,

Figure 2.4 shows this process for a small example network. For this process to successfully complete, it is necessary that the input degree sequence must sum to an even number, otherwise the connection process would terminate with a single unconnected stub. Furthermore, the configuration model will naturally produce a multi-graph: it is possible when selecting pairs of stubs that two stubs incident to the same node are selected, forming a self-edge, or that the stubs incident to a pair of nodes that already share an edge are selected, forming a multi-edge. If one wishes to construct a simple graph, that is $A(i,i) = 0$ and $\max\{A(i,j)\} = 1$ for every $i, j \in (1, 2, \ldots, N)$ there are three possible courses of action:

1. *The matching algorithm*: if a pair of stubs is selected that would result in a multi- or self-edge disregard them and make a new selection [50]. This method will result in a simple graph that preserves the input sequence but is known to produce biased

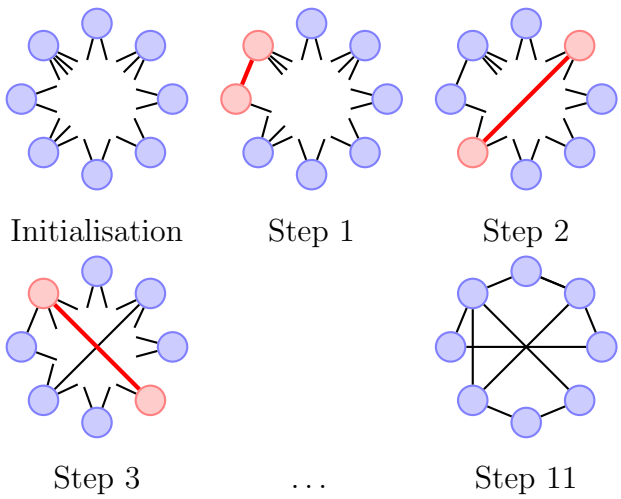


Figure 2.4: An example of the configuration model procedure on a small network of 8 nodes. A network is initialised with a number of nodes and each node is incident to a number of stubs that is specified by the input degree sequence. The algorithm proceeds to pair stubs at random to form edges. The process is complete once there are no remaining stubs.

results, i.e., does not sample uniformly from the space of all possible configuration model networks of given degree sequence [39].

2. *The refusal algorithm*: if a pair of stubs is selected that would result in a multi- or self-edge start the entire connection process from anew. This method yields no bias, will result in a simple graph and preserves the input sequence. However it will result in prohibitive running times for networks with higher average degree [39].

3. *The deleted configuration model*: at the end of the procedure delete any self-edge and collapse multi-edges down to a single edge. However, this will result in a network with a distribution of edges that does not precisely reflect the input degree sequence.

One of the great strengths of the configuration model is its tractability; it is relatively straightforward to *estimate* the probability of problematic edges occurring and show that they form a minuscule proportion of all edges for large networks. This result is essential when mitigating against the configuration model generating multi-graphs.

To estimate the number of multi-edges consider node i with degree k_i. One may assume without loss of generality that $k_i > 0$ since degree zero nodes can be disregarded from the network. In a network of $2m$ stubs the probability of a single stub from node i connecting to another stub incident node j with degree $k_j > 0$ is $\frac{k_j}{2m-1}$. However, node i is incident to k_i stubs so the probability of node i being connected to node j is $\frac{k_i k_j}{2m-1}$. Using this information it is straightforward to estimate the probability that two edges connect the same pair of nodes, namely

$$P(A(i,j) > 1) = \frac{k_i k_j (k_i - 1)(k_j - 1)}{(2m-1)(2m-2)}.$$

Adding this quantity over all pairs of nodes in the limit of a large configuration model network yields the following

$$\lim_{N\to\infty} \frac{1}{2} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{k_i(k_i-1)k_j(k_j-1)}{(2m-1)(2m-2)} \tag{2.3}$$

$$
\begin{aligned}
&= \lim_{N\to\infty} \frac{1}{2} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{k_i(k_i-1)k_j(k_j-1)}{(2m)^2} \\
&= \lim_{N\to\infty} \frac{1}{2} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{k_i(k_i-1)k_j(k_j-1)}{(N\langle k\rangle)^2} \\
&= \lim_{N\to\infty} \frac{1}{2} \sum_{i=1}^{N} \frac{k_i(k_i-1)}{N\langle k\rangle} \sum_{j=1}^{N} \frac{k_j(k_j-1)}{N\langle k\rangle} \\
&= \frac{1}{2} \left(\frac{\langle k^2\rangle - \langle k\rangle}{\langle k\rangle} \right)^2,
\end{aligned}
$$

where the additional coefficient of 2 in the denominator is used to cancel the double count and the following were used

$$
2m = N\langle k\rangle, \quad \langle k\rangle = \frac{1}{N}\sum_{i=1}^{N} k_i, \quad \langle k^2\rangle = \frac{1}{N}\sum_{i=1}^{N} k_i^2. \tag{2.4}
$$

Therefore the number of multi-edges is constant with respect to network size. There is a similar calculation for self-edges. A node of degree k_i has a total of $k_i(k_i-1)/2$ unique stub pairings. Upon selecting a single stub from this node the probability of selecting a second stub also incident to this node is

$$
\begin{aligned}
P(A(i,i)=1) &= \frac{k_i(k_i-1)}{2(2m-1)} \\
\Rightarrow \lim_{N\to\infty} \frac{k_i(k_i-1)}{2(2m-1)} &= \lim_{N\to\infty} \frac{k_i(k_i-1)}{4m}.
\end{aligned}
$$

Adding this quantity over all pairs of nodes in the limit of a large configuration model network yields the following

$$
\lim_{N\to\infty} \sum_{i=1}^{N} \frac{k_i(k_i-1)}{4m} = \frac{\langle k^2\rangle - \langle k\rangle}{2\langle k\rangle}. \tag{2.5}
$$

Equations (2.3) and (2.5) both show that the number of self- and multi-edges in the network does not depend on network size but only on the first and second moments of the degree distribution. This result is important when one wishes to mitigate against the effects of self- and multi-edges using the matching algorithm or the deleted configuration model. With increasing network size, and therefore increasing number of total edges, the number of problematic edges vanishes proportionally compared to all edges.

2.2.3 Lattice and bipartite rewiring

Watts & Strogatz provided a pioneering approach to generating complex networks by noticing that certain lattices were highly clustered and that by rewiring only a few edges at random the average path length of the network drastically dropped, so called *lattice rewiring* [78]. Their model characterises networks that have a relatively small *average path length*, the average number of steps in all shortest paths between all possible pairs of nodes, with relatively high clustering. Such networks are referred to as *small-world* networks. This elegant model is far from anecdotal; the unique combination of traits of small-world networks results in networks that are both globally and locally efficient, as is observed in many biological and technological networks [40, 70, 78].

It is possible to perform a similar process but initialised with a *bipartite network* as opposed to a lattice. A bipartite network, such as that shown in the top of Figure 2.5, is a network where two types of nodes form two disjoint groups. With populations often divided into groups, a bipartite network can be an appropriate choice of model, with one node class representing people and the other representing a group to which people belong [21, 55]. By representing each group with a complete subgraph, a larger network with non-zero clustering will form, as shown in Figure 2.5. To control the average degree, edges can be deleted from groups at random. These models represent some of the first network models based on realistic construction procedures. This work was the first to uncover that an increase in clustering corresponds to a network that becomes locally insular in structure. Using this model it was also subsequently shown that an increase in the global clustering coefficient corresponds to an increase in epidemic threshold, see [36], due to the increased local efficiency of the clustering [40]. However, whist the models can target a specific mean degree $\langle k \rangle$, they do not preserve the degree distribution in general [36].

2.2.4 Clustered random networks

Just as the configuration model represents a generalisation of the ER model, clustered random graphs may be considered a generalisation of the configuration model. It is characteristic of these methods to require as input a desired global clustering coefficient

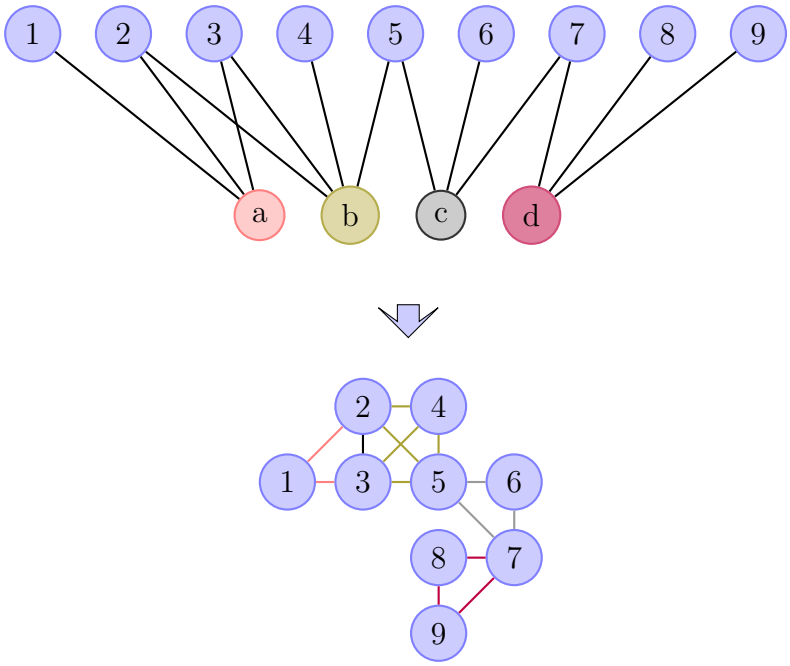


Figure 2.5: Creating a clustered network from a bipartite network. The four groups, $\{a, b, c, d\}$ are represented by the four subgraphs {{1,2,3}, {2,3,4,5}, {5,6,7}, {7,8,9}} respectively.

and/or a input degree distribution. The algorithm will then yield a network with the desired properties.

Whilst some of the first and subsequent attempts targeted only the global clustering coefficient at the cost of the degree distribution they still provided valuable insight into the impact of clustering on network structure and function [14, 21, 55, 58, 76]. It was again found that an increase in clustering corresponds to a network that becomes locally insular in structure. Nodes with higher local clustering were using edges to connect to 'friends-of-friends' rather than looking beyond their immediate neighbourhood, see Figure 2.6. This naturally impacts the size of the giant component [58] and, in turn, results in epidemics on clustered networks yielding fewer cumulative infected relative to networks with lower clustering [33].

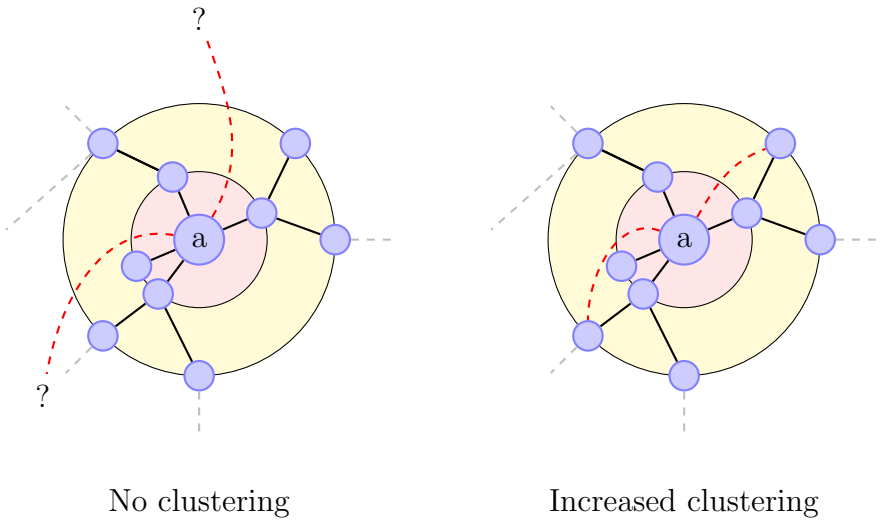


Figure 2.6: As the local clustering about node a increases, edges that would connect to the wider network must instead be used to connect to nodes that a is already relatively close to.

Models that target a given degree sequence and global clustering coefficient are fewer but do exist, see [68, 74] for examples. These models showed that increasing clustering may result in the network decomposing into small satellite sub-networks. This class of models was further enhanced by being able to specify how clustering aggregated

around nodes of a certain degree [68], uncovering the interplay between clustering and degree-degree correlations, i.e., positive and negative assortativity coefficients allowing for higher clustering and putting a tight bound on the highest possible clustering respectively.

2.2.5 Rewiring algorithms

Rewiring algorithms operate by selecting edges severing them, resulting in stubs, and reconnecting the stubs in a different configuration, so called edge swapping. This process is then repeated until a desired network property has been reached. These algorithms are applied to pre-constructed networks. At the most basic level one could select a random edge for deletion then select a random pair of nodes and form a new edge. However, this process would destroy the degree distribution. Instead, if two randomly selected edges are cut and the resulting stubs rewired to form two new edge pairs, the degree distribution will be preserved. However, this process process will resolve assortativity, or associativity, and clustering back to zero. Another consideration is the running time of such a process. For example, randomly swapping edges to greatly increase clustering network may take a considerable amount of time. This can be remedied with a Metropolis approach whereby the edges are rewired only if it results in an increase of the global clustering coefficient [35]. This algorithm has subsequently been improved by selecting certain arrangements of nodes and rewiring only specific edges so that: (1) the degrees were preserved and (2) a triangle would be more likely to be introduced [6, 24]. Such approaches are heavily relied upon and often form the basis for comparisons of clustered networks [20, 65–67].

More recently, a rewiring scheme has been proposed that preserves not only assortativity but also correlations between subgraphs of three or more nodes [43]. If one selects two edges, $A-B$ and $C-D$ so that two of the four nodes, A and D, belong to the same degree class then the following rewiring, $A-C$ and $B-C$ preserves the degree correlation. The above scheme is referred to as $2K$-preserving rewiring and is motivated by the dK-series, a series of families of distributions that specify the preponderance of non-isomorphic subgraphs of size d. For example, the $0K/1K/2K$-distributions cap-

ture average degree, degree distribution and assortativity respectively. Jamakovic *et al.* show how the above rewiring scheme is extended to $3K$-preserving rewiring [30]. These rewiring schemes are then applied to real-world networks to randomise the networks' structure beyond that of triangles. The authors also provide dK-targeting algorithms, that re-wire networks to increase a given property rather than anneal it. The authors make the important finding that the family of $3K$-distributions is sufficient to not only capture many of the classical network metrics but also determine much of the motif structure for motifs composed of 4 nodes. A number of open questions remain, such as: (*i*) do $3K$-distributions reveal any information regarding motifs of 5 nodes or more, (*ii*) what is the impact of $3K$-distributions on subgraphs that are not classified as motifs and (*iii*) what is the impact of all of this on network function.

2.2.6 Isolating higher-order structure

A network's degree distribution and clustering are key descriptions of network structure. Therefore for any network generation algorithm to support an investigation into higher-order structure, it must allow to vary the higher-order structure whilst keeping these canonical descriptions fixed. The work described in this thesis achieves this by building networks using the same degree distribution and global clustering coefficient but constructed using different subgraphs as building blocks. This style of building networks is not new [32, 58, 76], however, it is still not clear to what extent such differences impact on network function. In this work, impact of higher-order structure on network function will be tested using epidemiological models such as contagion processes.

2.3 Epidemic models

2.3.1 Compartmental models

Mathematical epidemiology is the study of infectious diseases in populations using mathematical modelling. Mathematical expressions are derived to, at least partially, capture the behaviour of an epidemic spreading through a population, such as the number of infected at a given time. *Compartmental modelling*, introduced by Kermack

and McKendrick in 1927, represents one of the first of such models [34]. More than this, it provides the blueprint for much of the contemporary work on this subject.

The problem may be stated thus: a single infected/contagious individual is introduced into a population of entirely susceptible individuals. When the infected individual meets a susceptible individual the pathogen has a chance to spread. If a susceptible individual catches the illness they become infected and so, in turn, can spread the disease. After a certain amount of time infected individuals recover, they no longer are contagious and are removed from the population [34]. Kermack and McKendrick proposed that a population may be *compartmentalised* to help characterise this phenomenon, i.e., divide the population into three classes: Susceptible (S), infected (I) and recovered (R). The idea is to form equations for $S(t)$, $I(t)$ and $R(t)$, the number of individuals in each compartment at time t. However, some simplifying assumptions are needed:

1. The time scale of infection is considerably smaller than the life-span of the individuals in the population. We therefore assume that $N = S(t) + I(t) + R(t)$, i.e., the population is closed.

2. Infected individuals transmit infection at a rate β.

3. Infected individuals recover independently at rate γ.

4. Each individual is equally likely to meet any other individual.

These assumptions allow one to write

$$\begin{aligned} \frac{dS}{dt} &= -\beta \frac{S(t)I(t)}{N}, \\ \frac{dI}{dt} &= \beta \frac{S(t)I(t)}{N} - \gamma I(t), \\ \frac{dR}{dt} &= \gamma I(t). \end{aligned} \tag{2.6}$$

Many variations of this system exist. Most commonly used are susceptible-infected susceptible (SIS) and susceptible-infected SI models where infected individual recover back into the susceptible state or remain perpetually infected respectively. In this thesis both SIS and SIR dynamics are used extensively.

Whilst compartmental modelling is generally very powerful, this first incarnation is limited in application due to the 4^{th} assumption listed above. The network equivalent of this assumption is to model the population's structure by a complete network. This may be plausible for modelling animal populations that are contained in pens but it is clearly an unrealistic assumption for populations in general. However, networks provide a highly malleable approach to modelling population structure in the context of epidemics.

2.3.2 Epidemics on networks

Theoretical approaches

There are two ways in which compartmental modelling may be implemented alongside a network model. Theoretically, the equations are somehow modified to account for network structure. Computationally, Sections 2.1 and 2.2 have described how to characterise and generate networks. All that remains is to select an appropriate method to simulating epidemic dynamics on the networks. This shall be addressed in the subsequent section.

For any given network there is a theoretical and exact way of describing the evolution of the epidemic. This is achieved by writing out the exact Kolmogorov or master equations that describe the probability of the system being any one of all of its possible states. Whilst this sounds ideal it requires 2^N and 3^N, where N denotes population size, equations for SIS and SIR epidemics respectively. Hence, this is unsuitable for realistic population sizes and a different approach is required. Fortunately this problem has received much attention and there is a large arsenal of theoretical approaches, including: moment closure approximations [24, 28, 33, 37, 73], effective degree [41, 44], heterogeneous mixing [22, 51] and *probability generating function* (PGF) based approaches [45, 54, 75, 76].

The PGF based approach to this problem, first introduced by Newman [54], and further developed by Volz, see [75], is particularly relevant to Chapter 4 of this thesis. Using the PGF of the degree distribution it is possible to describe both the connectivity of a randomly selected node and that of its neighbours. The connectivity of a randomly

selected nodes' neighbour may be derived from the PGF and is referred to as the *excess degree distribution*. For an example of this, please refer to Chapter 4. Using this approach, Karrer and Newman later showed how it may be generalised to include arbitrary subgraphs [32]. This result is critical to the third chapter of this thesis where we show that it is possible to derive an approximate ordinary differential equation (ODE) for epidemics that run on networks composed of arbitrary distributions of subgraphs.

Validation of the model is typically done by developing a network generation algorithm followed by simulating epidemics on the generated networks and by comparing these to results from the ODE model which needs to be able to incorporate and reflect the structure of the network.This is a critical property of the ODE model and the more complex the structure of the network the higher dimensional the ODE model.

Simulation of epidemics

Throughout this thesis, the *Gillespie algorithm* is used to simulate epidemic dynamics [17, 18]. To simulate an SIR epidemic following this algorithm perform the following steps:

1. Determine the parameters; set values for the per-link rate of infection, τ, and the rate of recovery γ.

2. Initialization; initialise the population as susceptible and from this population selected an initial infectious seed, i.e., $S(0) = N - I(0),\ I(0) = I_0 :\ 0 < I_0 \ll N,\ R(0) = 0,\ T(0) = 0$.

3. Create a list of all possible infectious events; refer to the adjacency matrix to determine the number of edges that connect a susceptible node to an infectious node. Represent each element in this list by the associated per-link rate of infection, τ.

4. Create a list all possible recovery events; each infected node appears once in this list and is represented by the associated recovery rate γ.

5. Determine the time until the next event; sum all rates associated with all events into a single rate and then use this rate as the parameter in an exponential

distribution to determine the time until the next event, t. Update: $T(n+1) = T(n) + t$.

6. Monte Carlo step, determination of the next event; select an event at random but proportional to the total rate associated with that event if the next event is:

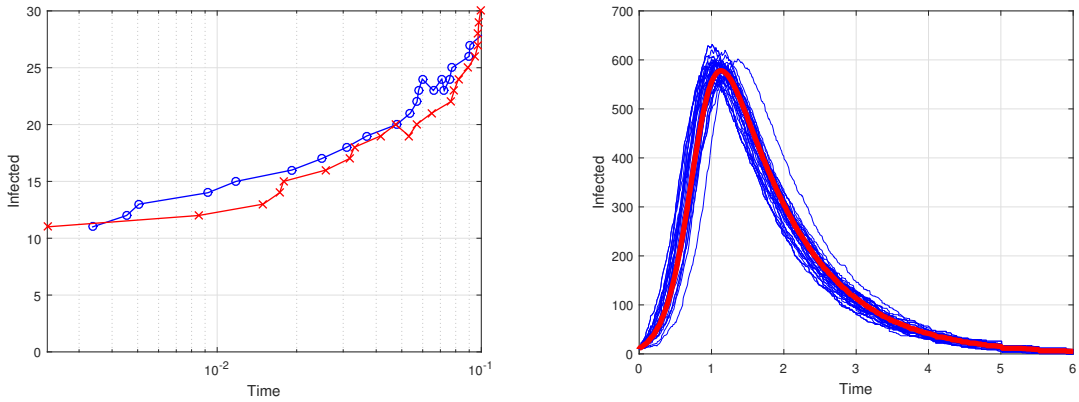
   (a) an infection; update $S(n+1) = S(n) - 1$, $I(n+1) = I(n) + 1$. Then update the number of susceptible-infected links and corresponding rates,

   (b) recovery: update $I(t+1) = I(t) - 1$ and $R(t+1) = R(t) + 1$. Then update the list of rates of infectious as well as recovery events.

7. Return to step 5 and repeat this process until there are no remaining infected individuals.

This algorithm can be easily modified to simulate an SIS epidemic by modified steps 6.b to $I(t+1) = I(t) - 1$ and $S(t+1) = S(t) + 1$. The above algorithm will yield 4 vectors: S, I, R and T that denote population level counts for the susceptible, infected and recovered populations as well a vector that marks the time at which event took place. Note that this algorithm is: (1) asynchronous, i.e., non-uniform time steps (2) continuous in time and (3) fully stochastic. In this thesis, to eliminate the effects of stochasticity, the average population counts of many simulations is taken. However, because the time steps are non-uniform the raw output must be further processed into uniform times steps before averages can be taken. Figure 2.7 shows example output from the Gillespie algorithm.

The merit of being able to capture the average of many stochastic simulations of an epidemic on networks with tractable ODE or other models is that it allows us to develop some deeper analytical understanding of the relationship between network characteristics and disease parameters. This is particularly revealing when looking at epidemic threshold, final epidemic size and endemic equilibrium. Epidemic models have been used widely to unravel the impact of different network characteristics. Such models continue to play an important role in understanding the impact of finer network structure by simply comparing epidemics simulated on what are believed to be similar,



Raw output Cloud plot

Figure 2.7: Two plots displaying output based on the Gillespie algorithms. The left hand figure shows the raw output of the Gillespie algorithm with its irregular time intervals. The right hand figure shows many individual simulations and their average, plotted in blue and red respectively.

or otherwise, networks. We use this method throughout the thesis to investigate the role and impact of higher-order structure on dynamics unfolding on networks.

Chapter 3

Paper I: Higher-order structure and epidemic dynamics in clustered networks

Martin Ritchie [1], Luc Berthouze[2,3], Thomas House[4] & Istvan Z. Kiss[1]

[1]School of Mathematical and Physical Sciences,
Department of Mathematics, University of Sussex,
Falmer, Brighton BN1 9QH, UK.
[2] Centre for Computational Neuroscience and Robotics,
University of Sussex, Falmer, Brighton BN1 9QH, UK.
[3] Institute of Child Health, London,
University College London, London WC1E 6BT, UK
[4] Warwick Mathematics Institute,
University of Warwick, Gibbet Hill Road, Coventry, CV4 7AL, UK
Journal of Theoretical Biology - 2014

3.1 Introduction

In this chapter I aim to go beyond the traditional approach to clustering, open and closed triples, and give a more comprehensive description of networks in terms of higher-order structure frequency (specifically order-four structures) and their distribution around nodes. In particular, I will examine existing clustered network generating algorithms with respect to their ability, or otherwise, to control higher-order network structure which sometimes may be regarded as a by-product of generating low-order structure that can preclude a correct interpretation of the impact of clustering. The chapter is structured as follows. We first introduce and describe a set of clustered network generating algorithms. I follow with a presentation of the network metrics (including a description of the motif identifying/counting algorithm) that I propose to quantify similarities and differences between the generated networks. I then analyse and discuss the impact of higher-order structural differences, at identical degree distribution and equal clustering, on SIS and SIR epidemics. Finally, I discuss how we motif-counting results and newly proposed measure for higher-order structures could be used to parametrise pairwise-like models with closure at the level of quadruples.

3.2 Material and methods

3.2.1 Network construction

A significant part of network research relies on networks with arbitrary degree distributions built using the configuration model [58]. This algorithm generates networks where nodes mix at random and where the probability that two nodes are connected is simply proportional to the product of their degree. Such networks coupled with stochastic node dynamics such as SIS, SIR or neural dynamics, are amenable to developing macroscopic low-dimensional ODE models that are in excellent agreement with values obtained from stochastic simulations. By construction, these networks are cycle free in the limit of large network size. While such networks can be considered in many cases as realistic or plausible models of some real-world networks, there are many instances

where networks have a high degree of structure that typically involves clusters of well connected nodes. Classic examples come from household models used in epidemiology [3, 5], and networks of social interactions in general. Motivated by this, there are a series of theoretical or synthetic network models that can be tuned to display increased levels of clustering [6, 14, 32, 58, 64, 76], where clustering denotes the ratio of closed loops of length three with respect to all possible open triple, irrespective of whether they are closed or not.

The classic algorithms to generate networks with tunable clustering include: (a) the spatial algorithm by Reed et al. [64], (b) an iterative method proposed by Eames [14], (c) a configuration model that includes clustering [32] and (d) the Big-V rewiring algorithm [6, 26]. In a recent study, Green et al. [20] showed that even under identical degree distributions and equal levels of clustering, networks built based on different algorithms can display a markedly different 'higher-order structure'. While their analysis identified large scale structural differences amongst networks with identical degree distribution and clustering, it did not consider extending the concept of clustering involving three nodes to higher-order structures with four or more nodes. The concept of motifs is not new [27, 32, 33, 71, 76] and understanding network structure through higher-order motifs is going to provide a level of detail which cannot be articulated by open or connected triples alone. Below I provide a brief description of the clustered network construction algorithms used in this chapter.

Big-V rewiring

The 'Big-V' is an iterative rewiring algorithm that can introduce clustering into any given network and is commonly used by network scientists [6, 20, 26]. At each iterative step, a chain of 5 distinct nodes (u-v-w-x-y) is selected at random and a clone network is generated where the links (u-v) and (x-y) are broken and the edges (u-y) and (v-x) are created. This leads to a single chain of 5 nodes being broken into a triangle and a disconnected pair, see Fig. 3.1. Local clustering for each node in the chain, as well as all of its neighbours, is computed in both the original and cloned networks and the new configuration is kept only if the level of clustering has increased. This process is repeated until the global clustering coefficient has reached the desired level.

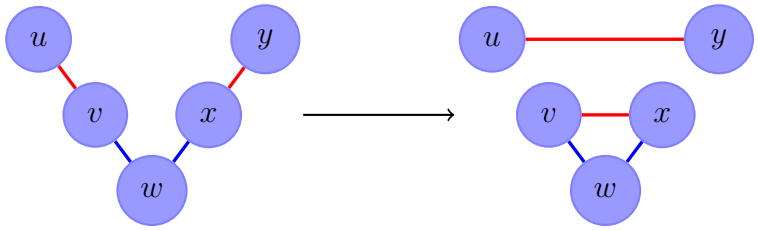


Figure 3.1: A single Big-V rewiring. (a) Identify a chain of 5 nodes with 4 edges and (b) if edges (u-v) or (x-y) are already part of a triangle the cuts will not be made, otherwise rewiring is performed, and (c) independently of the outcome of (b) the algorithm will proceed to find a new chain.

Motif decomposition rewiring

MD (Motif Decomposition), contributed by Thomas House, is an iterative rewiring algorithm that starts with a collection of complete subgraphs that are disconnected from one another and rewires edges randomly to reduce the clustering from its maximal value of 1 to the desired level. The following steps are performed:

i. initialise a network that is composed of m complete graphs each with n members so that: $N = nm$ and $\langle k \rangle = n - 1$,

ii. categorize every edge as 'local',

iii. for the first step only, select at random two local edges, cut them, and swap the stubs to form new edges. Mark the pair of new edges as global,

iv. select a local and a global edge, cut them, and swap the stubs to form new edges. Mark the pair of new edges as global,

v. check the global clustering, if the desired level has not been achieved repeat step (iv).

Fig. 3.2 illustrates this process being performed on a complete motif with 4 members. It should be noted that this method may work with a heterogeneous degree distribution in which case the network would need to be initialised with motifs of $k + 1$ nodes for each different degree k. MD has the significant advantage that it is computationally

cheap and that, in the limit of large networks, network properties can be calculated analytically (see appendix 3.5.1).

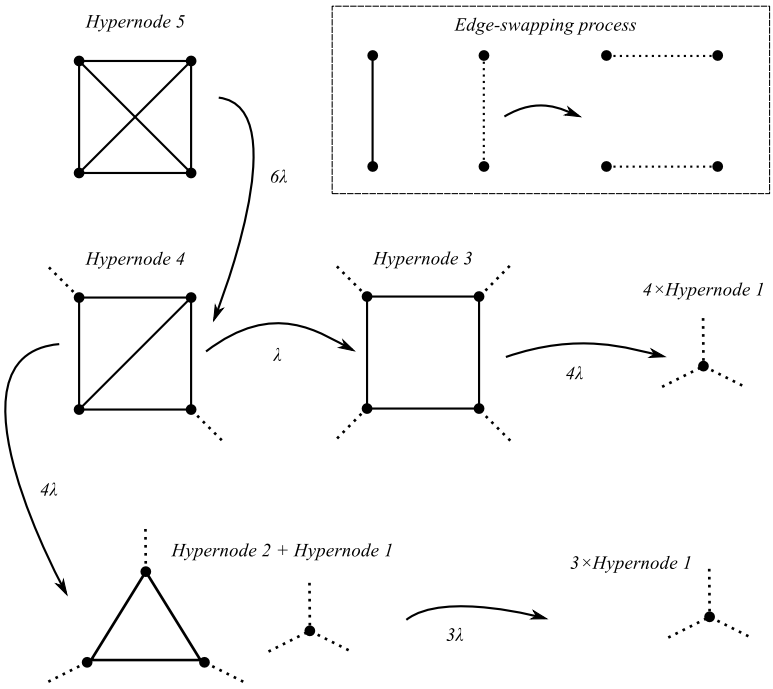


Figure 3.2: MD hyper-node configurations. The different possible hypernode configurations starting with a complete square. The process transitions between hypernode configurations at rate λ, which is included for clarity and may be set to one without loss of generality.

Clustered Configuration Model (CCM)

It is possible to modify the configuration model so that it constructs networks using specified motifs. Karrer et al. [32] and Volz et al. [76] have shown how to build networks using a configuration model that includes triangle motifs. This idea may be easily extended to allow for larger and more exotic motifs to be included in the networks' construction. Rather than just lines, the number of lines and corners of motifs that originate from a node can be varied. In any given motif a node can be considered as a corner and the number of stubs originating from this node that join it to the motif

defines its corner type, essential in describing corners of asymmetric structures. To generate a network using this method, the following steps are performed:

1. allocate to a node a number of stubs following a given degree distribution,
2. multinomially determine the configuration of subgraphs around nodes,
3. create lists for each corner type where a node that is allocated κ corners of a certain type, will appear κ times in the corresponding corner list,
4. draw corners at random and without replacement from the appropriate lists and connect with other corners to form motifs,
5. repeat until all lists are empty.

Fig. 3.3 illustrates corner allocation for an example node. Due to the nature of the configuration model self loops and double loops may be formed. The expected number of such occurrences depends only on degree, becoming negligibly small in the limit of large networks [57].

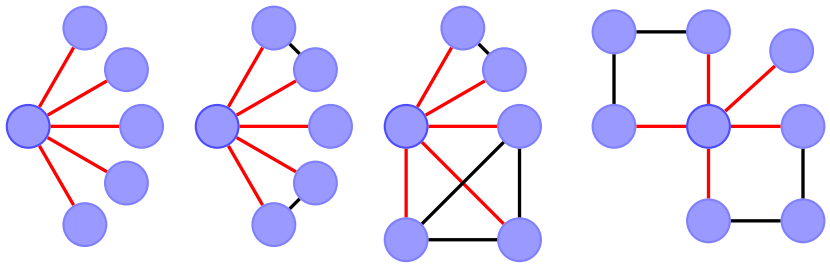


Figure 3.3: Corner/edge allocation. A node is initially allocated a quintuple of stubs. With probability p_1, p_2, p_3 and p_4 the node will be part of a number of different structures as shown above. In this work homogeneous networks where each node is incident to five edges have been used. If a different degree or degree distribution is required then the configuration of motifs will need to be adjusted accordingly.

	Lines	Triangles	Complete squares
$\phi = 0.2$	2.5	0.5	0.5
$\phi = 0.4$	0	1	1

Table 3.1: The expected number of lines, triangles and complete squares per node for each level of clustering used.

In this chapter homogeneous, where each node is incident five edges, CCM networks are used with clustering of $\phi = 0.2$ and $\phi = 0.4$. The stub configurations to generate such networks are as follows:

1. $\phi = 0.2$: with probability $p_1 = 0.5$ the quintuple of stubs is maintained as independent links, and with probability $p_2 = 1 - p_1$ the quintuple is arranged into one complete square corner and one triangle corner,

2. $\phi = 0.4$: every node is allocated one complete square corner and one triangle corner,

3. $\phi = 0.8$: as this algorithm does not allow overlaps between motifs, this value of clustering cannot be achieved.

Table 3.1 shows the expected motif allocation per node. The configuration model allow us to analytically determine some of the networks structural properties, more specifically the PGF (Probability Generating Function) of the degree/motif distribution.

The CCM algorithm for this work was configured as follows. First, initialise each node with five stubs. Then let p_1 denote the probability that the five stubs form lines, p_2 two triangles and one line, p_3 one complete square and one triangle and p_4 two empty squares and one line. The probabilities are chosen such that $\sum_i p_i = 1$. Let x_i denote the dummy variables of the PGF that corresponds to corner types: x_1 (simple stubs), x_2 (two triangles and a simple stub), x_3 (a complete square and a triangle), x_4 (two empty squares and a simple stub). The PGF of the networks degree/corner distribution may now be written as:

$$\Psi(x_1, x_2, x_3, x_4) = x_1^5 p_1 + x_1 x_2^2 p_2 + x_2 x_3 p_3 + x_1 x_4^2 p_4, \tag{3.1}$$

and the original stub distribution may be recovered by substituting each x_i with x_1^n where n is the corner-stub cardinality:

$$\begin{aligned} \psi(x_1) &= x_1^5 p_1 + x_1 (x_1^2)^2 p_2 + x_1^2 x_1^3 p_3 + x_1 (x_1^2)^2 p_4 && (3.2) \\ &= x_1^5 (p_1 + p_2 + p_3 + p_4) = x_1^5. && (3.3) \end{aligned}$$

$\psi'(1)$ yields the expected degree, and $N\psi''(1)/2$ yields the number of paths of length three in the network [76]. The number of unique triangles in the network can be determined by Ψ:

$$[\triangle] = N \left(\frac{\Psi_{x_2}(1,1,1,1)}{3} + \frac{4 \cdot \Psi_{x_3}(1,1,1,1)}{4} \right), \tag{3.4}$$

since each square is quadruply counted and contains four separate triangles. Clustering is measured as the ratio of three times the number of triangles to all closed and unclosed triples:

$$\begin{aligned} \phi_{global} &= \frac{3N \left(\frac{\Psi_{x_2}(1,1,1,1)}{3} + \Psi_{x_3}(1,1,1,1) \right)}{N\psi''(1)} && (3.5) \\ &= \frac{\Psi_{x_2}(1,1,1,1) + 3\Psi_{x_3}(1,1,1,1)}{\psi''(1)} && (3.6) \\ &= \frac{p_2 + 2p_3}{5} && (3.7) \end{aligned}$$

For the two types of CCM networks used in this study: $p_1 = 0.5,\ p_3 = 0.5$ yields $\phi = 0.2$ and $p_3 = 1$ yields $\phi = 0.4$ (see table 3.1).

3.2.2 Network metrics: Third and higher-order network structure

Here I give a succinct summary of the classic and newly proposed network metrics that will be used to compare and contrast the networks resulting from the different algorithms. Although the novelty of the chapter is around order-four structure, I will first consider classic (or third-order) network measures, such as clustering in the global sense as well as distribution of clustering at node level, nodal betweenness centrality, and connected component analysis via percolation. I then augment the classic network

descriptions with an analysis of the distribution of motifs of order higher than closed and open triples both globally and on a per node basis. A network of N individuals is represented with an adjacency matrix, $A \in \{0,1\}^{N^2}$. A pair of individuals (i,j) share a connection if $A_{i,j} = 1$. The networks are undirected, $A = A^T$, and self loops are not allowed $A_{i,i} = 0,\ \forall i \in N$.

Clustering

Clustering may be defined in two ways [78]: Local (node level) and global (network level). The local clustering of a node n, of degree n_k, is the ratio of connections between neighbours of n and potential connections of neighbours of n. Let $\mathcal{N}$ denote the sub-adjacency matrix of the neighbourhood of n then:

$$\phi_{local} = \frac{\sum_{i,j} \mathcal{N}_{i,j}/2}{n_k(n_k-1)/2}. \tag{3.8}$$

Global clustering is defined as the ratio of the total number of closed triples to the total number of connected structures with 3 nodes. This may be computed from the adjacency matrix as [33]:

$$\phi_{global} = \frac{trace(A^3)}{\|A^2\| - trace(A^2)}, \tag{3.9}$$

where $\|A^2\|$ denotes the sum of all elements of A^2. Manipulating the adjacency matrix in this way yields multiplicative counts. An alternative method to obtain the equivalent counts is as follows:

$$[\vee + \triangle] = \sum_{i,j,k,\ i\neq j\neq k} a_{i,j}a_{j,k}, \tag{3.10}$$

yielding all connected structures of 3 nodes (closed and unclosed), similarly

$$[\triangle] = \sum_{i,j,k,\ i\neq j\neq k} a_{i,j}a_{i,k}a_{j,k}, \tag{3.11}$$

yielding six times the number of unique triangles. A more complete description of this approach is provided in Appendix 3.5.3, along with a conjecture of a possible mapping between unique and multiplicative counts.

Nodal betweenness centrality

Nodal betweenness centrality measures how often a node appears in the set of shortest paths (which I shall denote s), geodesics, of the network [16]. Nodes with high betweenness centrality will more frequently appear in shortest paths than low ranked nodes. The betweenness centrality of a node n can be computed by:

$$B_{bc}(n) = \frac{\sum_{i \neq j \neq n} s_{i,j}(n)}{|s_{i,j}|}, \tag{3.12}$$

where $s_{i,j}(n)$ denotes the number of shortest paths from i to j that contain node n. The removal of nodes with high betweenness centrality can significantly affect the flow of dynamical processes on the network [57].

Connected component analysis

CCs (Connected components) are sets of nodes where any node may be reached from any other node that is a member of the set. CCs are used to describe the macroscopic structure of a network, as opposed to clustering which describes the local structure of the network. Highly clustered networks contain many components that are weakly connected to, or disconnected from, one another. It has previously been shown [20] that the GCCs (Giant Connected Component) of highly clustered networks are composed of many CCs of varying size and that removing a low proportion of edges can be enough to isolate parts of the network. To perform the analysis: I generate a list of all edges in a network, cycle through each edge in the list and remove it with probability p_r, compute the size and frequency of all components remaining, and plot the cumulative distribution of component size.

Motif frequency and distribution

Clustering (local or global) essentially measures the occurrence of triangles in a network. It does not distinguish two separate triangles from two triangles that share an edge, neither can it describe cycles of order-four or larger. From the perspective of characterising higher-order structure it is a very coarse measurement. In this chapter

all cycles of order-four i.e. empty squares, diagonal squares, and complete squares motifs, see Fig. 2.3 are considered at both network and node levels. It is possible to define new clustering type metrics using structures larger than triangles. Proceeding in a way similar to classic (third-order) clustering and limiting ourselves to 4-node structures connected in a loop, it is possible to define four new structural measurements: the ratio of unclosed quadruples ($1 - \phi_4^1$), 'empty' squares (ϕ_4^2), squares with a single diagonal (ϕ_4^3), and complete squares (ϕ_4^4) to all connected structures of 4 nodes. I present we results in two formats: (i) global ratios of *unique* order-four structure counts to all unique paths counts, closed and unclosed. (ii) probability distribution of finding x structures of a certain type associated with a given node. These measurements alongside clustering will provide a higher resolution analysis of network architecture. A brief synopsis of how to compute non-trivial paths of length $l + 1$ is as follows (see appendix 3.5.2 for the full pseudo-code, all path lengths refer to the number of *edges* and $A(\cdot, \cdot)$ is the adjacency matrix):

1. consider a path P of length l, and identify a head $H(P)$ (1^{st} node of the path) and a tail $T(P)$ (the last node).

2. For each neighbour n of $T(P)$, if (i) $A(H(P), n) = 1$, (ii) it has not already been counted as a closed path, and (iii) its reverse has not been counted as a closed path then count a closed path of length $l + 1$,

3. for each neighbour n of $T(P)$, if (i) $A(H(P), n) = 0$, (ii) it has not already been counted as an open path, and (iii) its reverse has not been counted as an open path then count an open path of length $l + 1$,

4. for all closed paths of length $l + 1$ remove circular and reverse circular permutations,

5. categorize each closed path by its completeness, i.e., the number (if any) of diagonals in a square.

3.2.3 Dynamics on networks

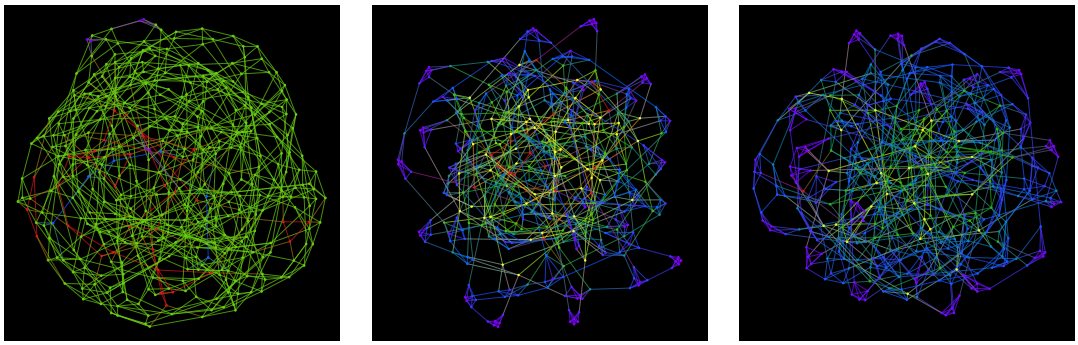
To establish the overall impact of higher-order network structure, simulations of various dynamics are performed on the generated networks. First, I use the Markovian SIR (susceptible-infected-recovered) model with a per-contact infection rate τ and recovery rate γ. All simulations are performed using the Gillespie algorithm [17] (see Chapter. 2. To assess the impact of loops and cycles, I also simulate the SIS epidemic which is more likely to highlight differences in the cycle/motif composition. I shall see that structural differences between networks with the same degree distribution and clustering manifest in epidemiological differences with regard to dynamics on the networks. Previous work [24] used Kirkwood's superposition approximation to predict the effect of order-four structure on epidemic dynamics. For SIS dynamics it was concluded that the presence of empty square structures reduces the endemic state for all levels of ϕ, complete squares may increase or decrease the endemic state, and that diagonal squares had very little effect on the endemic state. In the following I make comparisons between networks that use different distributions of order-four motifs. Networks with markedly different order-four motif distributions have been previously noted to produce different epidemiological behaviour, see [6, 24, 28].

3.3 Results

Using the various construction algorithms, I give an overarching analysis of structural differences between networks with the same degree distribution and same levels of classic clustering. All networks used are homogeneous with $\langle k \rangle = 5$, allowing for the formation of structures/loops while keeping the complexity to a manageable level. I carry out we analysis on a range of clustering values (i.e. $\phi = 0.2, 0.4, 0.8$) to measure and evaluate the extent to which clustering can emerge from, or determine, different configurations of order-four structures. The CCM is currently unable to produce networks with $\phi = 0.8$ for this particular degree distribution, so it is not represented for these parameters.

3.3.1 Overall feature and structure of the network

Gephi [8] was used to visualize sample networks generated by the proposed algorithms, see Fig. 3.4. In these figures nodes are colour coded according to their degree of clustering, with un-clustered nodes coloured with nuances closer to the red end of the spectrum, and more highly clustered nodes coloured with shades closer to the blue end of the spectrum. The figure clearly illustrates that the CCM algorithm gives rise to networks with an extremely homogeneous structure, whilst the rewiring algorithms (i.e. Big-V and MD) construct networks with more heterogeneity in clustering at node level. It is also evident that this difference translates into a more modular structure for the rewired networks. The CCM networks stand out as being structurally different from the networks generated by the other algorithms; as well as being homogeneous in degree, they are also homogeneous in structure.



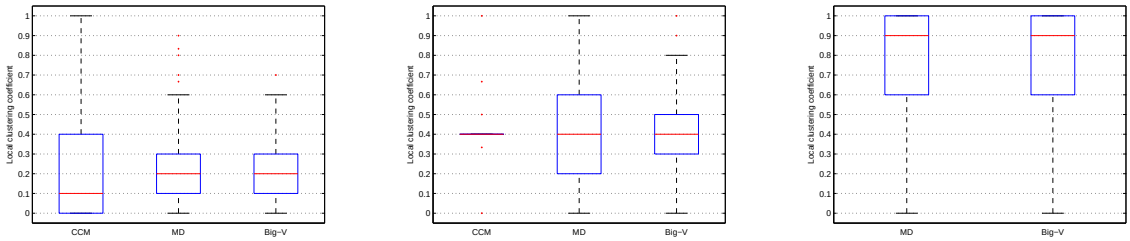
CCM MD Big-V

Figure 3.4: Example networks. Homogeneous networks with all parameters held equal, $N = 400$, $k = 5$ and $\phi = 0.4$. The nodes are coloured so those at the red end of the spectrum have low local clustering, and those at the blue end of the spectrum have high local clustering.

3.3.2 Distribution of clustering and centrality

The almost homogeneous distribution of the local clustering (see Fig. 3.5) and betweenness centrality (see Fig. 3.6) of the CCM networks is expected since by construction

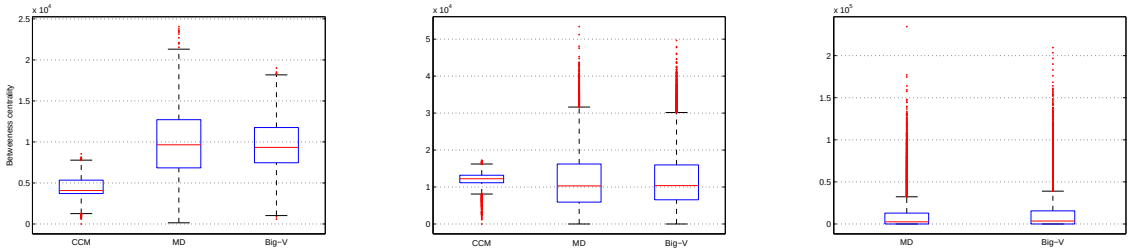
every node has the same local structure. For $\phi = 0.2$ I know that each node has a quintuple of stubs with probability p_1 or a complete square and a triangle with probability $p_2 = 1 - p_1$. When $\phi = 0.4$ every node is a member of one triangle and one complete square.



$\phi = 0.2$ $\phi = 0.4$ $\phi = 0.8$

Figure 3.5: Distribution of local clustering. Boxplots of local clustering measured from 20 homogeneous networks, $N = 1998$, $\langle k \rangle = 5$. The size of the network was chosen to be divisible by 6 due to the MD algorithm starting with disjointed fully-connected hexagons. Local clustering is a measure of interconnectivity between neighbours of a given node. CCM shows an extremely tight distribution of clustering for $\phi = 0.4$, as I would expect given that each node is allocated the same number and type of structure. MD provides the largest variance in local clustering. The CCM is currently unable to produce networks with $\phi = 0.8$ for this particular degree distribution.

The Big-V algorithm introduces clustering in more heterogeneous manner, with half of the nodes having clustering in the range $0.3 \leq \phi \leq 0.5$. The MD algorithm provides the largest spread of clustering with half of the nodes having clustering in the range $0.2 \leq \phi \leq 0.6$. The box-plot (Fig. 3.5) of local clustering shows the tendency of the MD algorithm to leave motifs unchanged. These complete motifs must be compensated with other parts of the network being decomposed into a much more random graph-type structure. The MD algorithm relies on random edge swapping to decompose the network into a more random structure. This will tend to result in at least a few leftover fully-connected motifs which are only destroyed at very low levels of clustering. Hence, when the frequency of such motifs is still large but the overall, desired, clustering is



$\phi = 0.2$ $\phi = 0.4$ $\phi = 0.8$

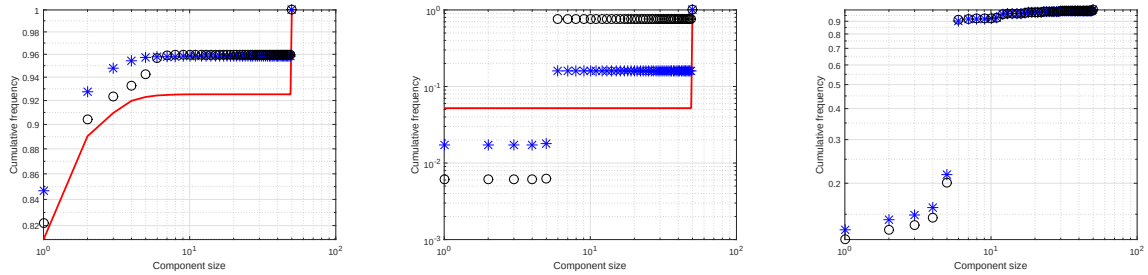
Figure 3.6: Distribution of betweenness centrality. Box-plots of nodal betweenness centrality measured from 20 homogeneous networks, $N = 1998$, $\langle k \rangle = 5$. Betweenness centrality ranks nodes on how often they appear in paths between other nodes. As clustering is tuned higher, the CCM and Big-V rewiring algorithms isolate fully-connected clustered components of $\langle k \rangle + 1$ nodes away from the GCC. At this level of clustering highly-connected sets of nodes are still weakly attached to the GCC, yielding the large number of outliers observed in the plot, and hence, a high spread of betweenness centrality values.

moderate, the connected parts of the network have to be left weakly clustered.

The plot of betweenness centrality (Fig. 3.6) illustrates this description in a more subtle way. Nodes embedded in a motif will have a low betweenness centrality whilst those that act as bridges between structures and the rest of the network will be more highly ranked. I observe that the betweenness centrality plot shows a slightly higher spread for MD networks. The removal of nodes (with a high betweenness centrality rank) is more likely to have a bigger impact on dynamics flowing on the network constructed by the MD algorithm.

3.3.3 Connected component analysis

Fig. 3.7 show that the resulting networks from the three different algorithms all exhibit significantly different behaviour in the connected components as edges are removed. For a well connected network with low clustering, low values of p_r leave the macroscopic structure of the networks unchanged, i.e., the entire network is contained within a single GCC. Networks with low clustering are resilient to the removal of a relatively small number of edges, this behaviour has been previously noted [20]. However, for $p_r = 0.4$ the network shows the CCM networks are the most resilient to edges being removed followed by the MD and Big-V.



$\phi = 0.2,\ p_r = 0.4$ $\phi = 0.4,\ p_r = 0.1$ $\phi = 0.8,\ p_r = 0.4$

Figure 3.7: Frequency of component sizes as edges are removed from the network with probability p_r. CMM, Big-V and MD networks are denotes with a solid line, stars and circles respectively. Results are taken from homogeneous networks with $\langle k \rangle = 5$ and $N = 1998$. CCM networks are not represented when $\phi = 0.8$.

	ϕ_4^1	ϕ_4^2 □	ϕ_4^3 ◪	ϕ_4^4 ⊠
CCM, $\phi = 0.2$	0.0053	0.0007	0.0001	0.0045
Big-V, $\phi = 0.2$	0.0117	0.0004	0.0104	0.0009
MD, $\phi = 0.2$	0.0239	0.0057	0.0149	0.0034
CCM, $\phi = 0.4$	0.0169	0.0003	0.0002	0.0164
Big-V, $\phi = 0.4$	0.0570	0.0010	0.0400	0.0160
MD, $\phi = 0.4$	0.0731	0.0083	0.0444	0.0204
Big-V, $\phi = 0.8$	0.3150	0.0013	0.0900	0.2237
MD, $\phi = 0.8$	0.3405	0.0044	0.1062	0.2299

Table 3.2: For each level of clustering the table has been sorted in ascending ϕ then ascending ϕ_4^1, where ϕ_4^1 gives the proportion of all closed quadruples. The above table is computed using unique counts.

Fig. 3.7 shows that for $\phi = 0.4$ removing a relatively few number of edges results in markedly different network topologies: the CCM networks result in many isolated single nodes compared to the Big-V and MD networks that result in isolated 6 node components. The MD networks yield the most isolated components as it is will likely preserve a more modular structure. As clustering is pushed higher, and despite this level of clustering tightly constraining network structure, the Big-V networks still show a appreciable differences to the MD networks upon the removal of edges.

3.3.4 Motif statistics for all network types

Table 3.2 shows that third-order clustering conveys little information about order-four motifs. The proportion of closed quadruples to all connected structures of 4 nodes increases with clustering, and for high levels of clustering there is a strong dependence on complete squares. However, the algorithms' lack of control of order-four structure is apparent at moderate levels of clustering ($\phi = 0.2,\ 0.4$) where there is no consistent presence of closed order-four structures across networks of equal clustering. The difference in ϕ_4^1 is due to triangles which do not share an edge; indeed these triangles are not

measured by this metric. The distribution of triangles is important at higher levels of clustering where they often share edges or overlap to form order-four structures.

Reading column-wise down Fig. 3.8, I see a more particular dependence on squares with diagonals as clustering increases. For $\phi = 0.4$ there is the greatest heterogeneity in the distribution of diagonal squares. When $\phi = 0.8$ I observe that diagonals can only appear in certain combinations about a node, following an almost tri-modal distribution. Again reading column-wise down, for complete squares I see a general trend of increasing complete square prevalence with increased clustering. Nodes may have a count of ten complete squares associated with them when they are members of a complete, and isolated, six-node structure. At all levels of clustering the probability of finding an empty square associated with a node is rare. This is because, despite being a structure of order-four, squares do not contribute to clustering.

Finally, Fig. 3.8 also reveals that networks generated by the Big-V algorithm contain empty square motifs with very low frequency. The algorithm searches for unclosed triples contained within strings of five nodes and closes them. Only motifs that may be constructed out of triangles can be expected in Big-V networks in any significant quantity. The MD algorithm also generates few empty square motifs and the CCM algorithm will only include them by specification.

3.3.5 Dynamics on the networks: evaluation and comparison

The effect of higher-order structure on epidemics is not obvious. For triangles it is observed that when an initially infected individual infects a second, the two infected nodes then compete for the same remaining susceptible. For empty squares and longer loops the effect is similar but less dramatic. Fig. 3.9 shows that the initial epidemic spread is slower for networks which exhibit loops. By opening a closed motif whilst preserving degree, two new individuals must be added so the effect of competition is inversely proportional to the motif size. Connectivity within the motif may also negate the effect of competition.

When simulating epidemics on networks with $\phi = 0.2$, the CCM networks show a slower spread of infection (Fig. 3.10). At this level of clustering the CCM algorithm

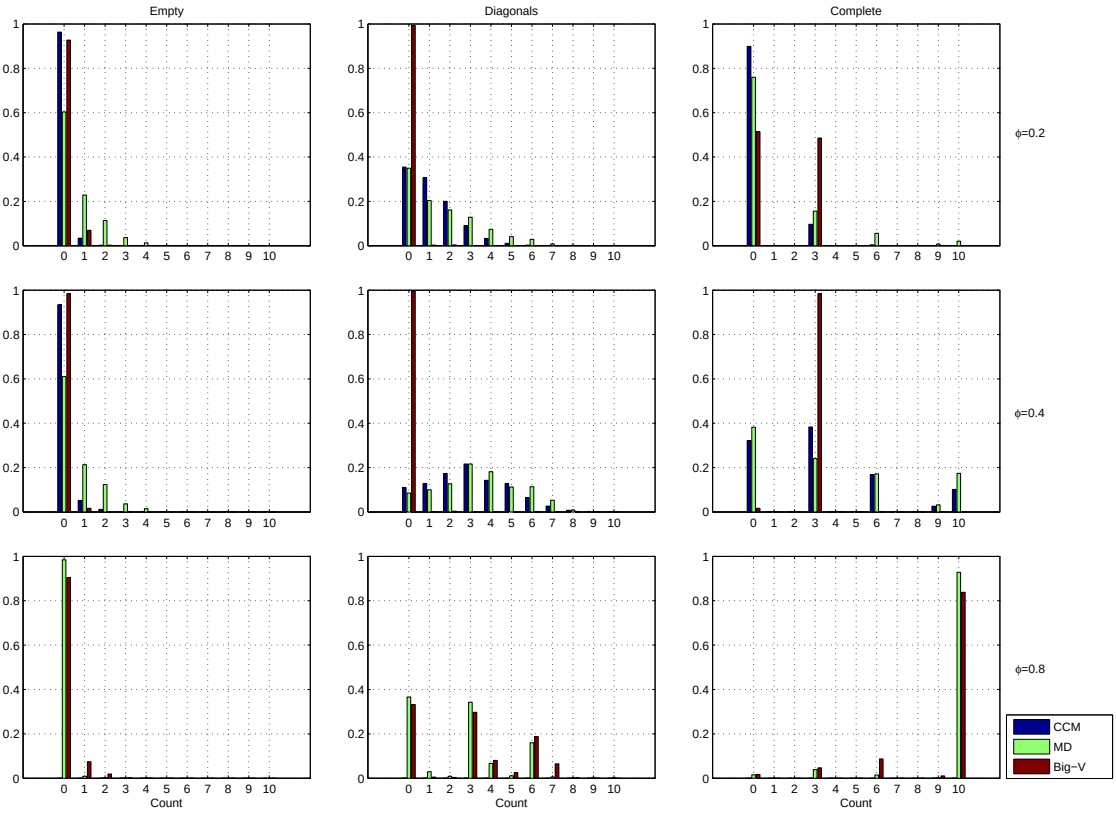
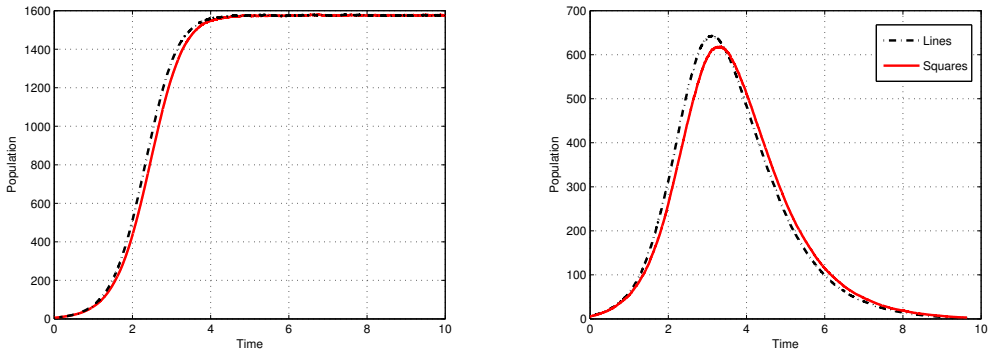


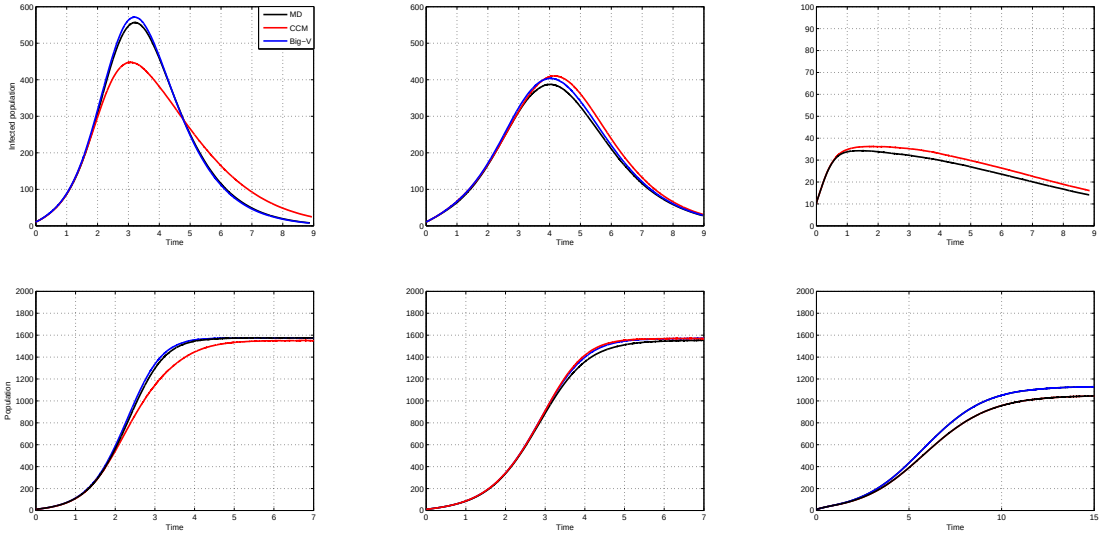
Figure 3.8: Order-four motif distribution. The per-node distribution of the number of unique counts of order-four motifs, for all previously used networks. See Appendix 3.5.2 which details how motifs are counted.



SIS SIR

Figure 3.9: Random vs square comparison. 20 homogeneous networks were generated with: $N = 1998$, $\langle k \rangle = 5$ and $\phi = 0.0018$. The plots correspond to averaging Gillespie simulations on each of the networks with parameters $\tau = \gamma = 1$, and 5 initially infectious nodes. The networks marked 'square' were constructed by allowing two squares to be formed out of a nodes 5 stubs, compared against a random network. The plots show comparisons between the prevalence of infection for SIS and SIR dynamics.

breaks a quintuple of stubs into all lines with $p = 1/2$, or a complete square corner and a triangle corner with $p = 1/2$. Thus, the CCM networks exhibit areas of high clustering in which the disease will spread more slowly than in areas of low clustering. At $\phi = 0.2$ the CCM networks exhibit a slower spread of infection for both SIS and SIR epidemics. Reading row-wise from left to right it is clear that higher levels of clustering slows the epidemic, see the difference between $\phi = 0.2$ and $\phi = 0.4$, with a less dramatic effect for SIS epidemics. Tuning clustering to an even higher level leads to the network breaking down into many disjointed components, such that connectivity within these is excellent. This means that the initial spread could be very fast, but this is quickly curtailed by limited or no connectivity between the highly connected components.

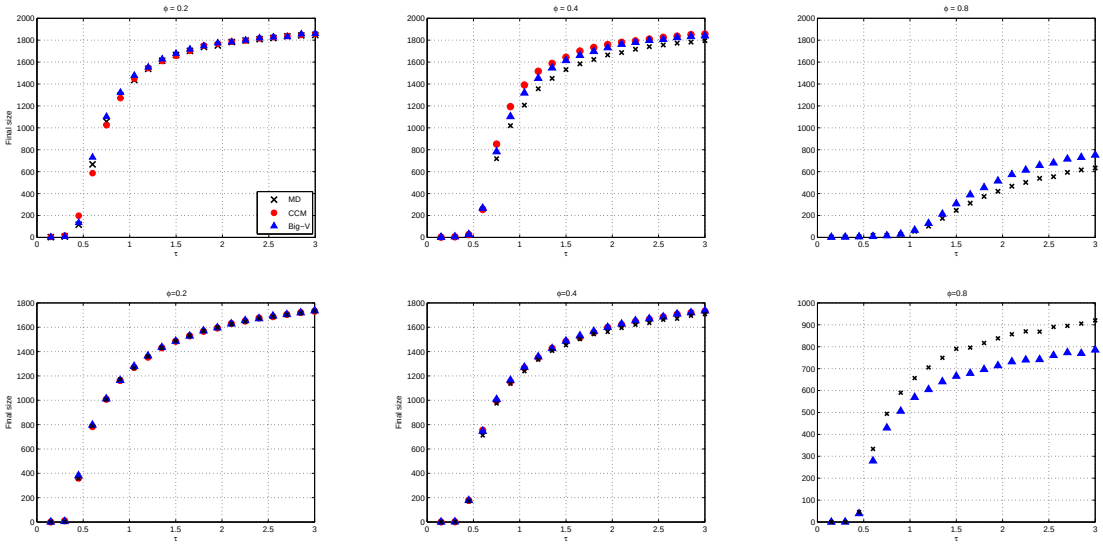


$\phi = 0.2$ $\phi = 0.4$ $\phi = 0.8$

Figure 3.10: SIR and SIS dynamics. 20 homogeneous networks were generated with: $N = 1998$, and $\langle k \rangle = 5$, and the results show the average of 100 Gillespie epidemics on each network realisation. The epidemics were run with parameters $\tau = \gamma = 1$, and were seeded with 5 infectious nodes. The top and bottom rows show the prevalence levels for a SIR and SIS epidemics, respectively. There are no results for the CCM networks when clustering is set to $\phi = 0.88$

The rewiring algorithms tend to produce networks that contain clustered motifs

that are poorly connected to the rest of the networks. Nodes with high betweenness centrality are important in SIR-type processes, which when recovered significantly hinder the propagation of the epidemic. All of the rewiring algorithms produce nodes with high betweenness centrality when compared to the CCM algorithm. It has previously been noted that the MD networks are particularly dependent on isolated motifs; this is reflected in consistently smaller final epidemic size. The CCM networks have a more consistent connectivity throughout the network yielding a greater final size (see Fig. 3.11).



$\phi = 0.2$ $\phi = 0.4$ $\phi = 0.8$

Figure 3.11: Plots of final epidemic size (top row) and endemic equilibrium (bottom row) for values of τ increasing from $\tau = 0.1$ to $\tau = 3$ in increments of 0.15. Twenty networks were generated, ten Gillespie simulations were performed for each value of τ. The networks were homogeneous with $\langle k \rangle = 5$ and $N = 1998$.

3.4 Discussion

Many network models operate on and use synthetic networks designed to be able to control and tune properties such as degree distribution, mixing, and global clustering.

However, as shown in this chapter, controlling only lower-order properties (specifically the degree distribution and global clustering) leaves the potential for diversity in the networks' higher-order structure and this must be taken into consideration when accuracy is required. This is demonstrated by the differences in structural metrics and resulting behaviour of dynamics across networks with equal degree distribution and global clustering coefficient. Moreover, differences were observed between the motif decomposition and Big-V networks despite them being sharing equal parametrised and, importantly, sharing very similar network construction philosophies.

In this study, I highlighted that synthetic algorithms that generate networks with tunable clustering do lead to different higher-order structures, such that networks with the same degree distribution and level of clustering can yield different dynamics on the networks. To help illustrate this point this chapter provides a motif counting algorithm that measures the number of motifs found in the networks. I found that all algorithms have their own unique order-four motif signatures for a given degree distribution and global clustering coefficient. I have also attempted to characterise such motifs with order-four transitivity ratios.

The measures I have proposed are ratios of the uniquely counted, closed motifs of order-four to the unique count of all connected structures of four nodes. This is conceptually convenient but these values may not be suitable for use in low-dimensional ODE approximations, such as the pairwise model. Global clustering is not defined using purely unique counts (see appendix 3.5.3) and yields a different value when the unique counts are used. In appendix 3.5.3, I hypothesise the correct counts of motifs and paths for use in clustering-type ratios. Whilst counting uniquely significantly reduces computational complexity, it has the slight disadvantage that it does not provide the multiplicative type of counting used in pairwise models. In the Appendix, I conjecture that this can be easily overcome by simply multiplying unique counts with the cardinality of the automorphism group corresponding to the motif.

In general, I expect that Big-V rewiring will be the most random way to introduce clustering without model-specific artefacts. However, this comes at what is occasionally prohibitive computational cost when high global clustering is desired. The clustered configuration model (CCM) is computationally cheap, and analytically tractable, but is

a long way from randomly introduced clustering. In this context, MD can be viewed as a computationally cheap and (given sufficient effort) analytically tractable alternative to Big-V that produces very similar network phenomenology.

It has been demonstrated that care needs to be taken when trying to extend modelling to clustered networks. While models for simple clustered networks composed of exclusively non-overlapping triangles and edges have been developed, it is going to be more challenging to extend to networks with more complex structures and motifs. Motifs such as a square with a diagonal or a fully connected square may fulfil some function depending on the area of application (e.g. genetic regulatory networks, cortical networks), and thus measuring and quantifying this correctly is crucial for further model development. Many natural extensions for this work exist which include considerations around higher-order structure, algorithm efficiency in measuring these and development of synthetic network models that allow robust and transparent control of not only lower, but also higher-order structures.

Acknowledgements

Martin Ritchie acknowledges funding for his PhD studies from EPSRC (Engineering and Physical Sciences Research Council) and the University of Sussex. Thomas House is supported by the EPSRC, and would like to thank Charo I. Del Genio for discussions on the Motif Decomposition algorithm.

3.5 Appendices

3.5.1 Motif decomposition, analysis

It is possible to write down the dynamics for this process in the limit of large networks by decomposing motifs into hyper nodes (considering each motif at a higher level as a node) and considering the links between them. By identifying a hyper-node with n edges as Q_n, it is possible to write equations for the normalised count of each hyper-node:

$$\frac{d}{dt}Q_5 = -6\lambda Q_5, \tag{3.13}$$
$$\frac{d}{dt}Q_4 = 6\lambda Q_5 - 5\lambda Q_4, \tag{3.14}$$
$$\frac{d}{dt}Q_3 = \lambda Q_4 - 4\lambda Q_3, \tag{3.15}$$
$$\frac{d}{dt}Q_2 = 4\lambda Q_3 - 3\lambda Q_2, \tag{3.16}$$
$$\frac{d}{dt}Q_1 = 4\lambda Q_4 + 16\lambda Q_3 + 9\lambda Q_2. \tag{3.17}$$

These equations can be solved for initial condition $Q(0) = (0, 0, 0, 0, 1/4)$ and evolve towards $Q(\infty) = (1, 0, 0, 0, 0)$. The rate λ is just included for clarity and can be set to 1 with no loss of generality. The process stops at a 'time' t^* when the desired level of clustering has been achieved:

$$\frac{1}{6}\sum_i T_i Q_i(t^*) = \phi, \tag{3.18}$$

where T_i denotes the number of triangles associated with each hyper-node type. The above equation may be solved for t^*, and inserted into the equations (1)-(5) to obtain a prediction for motif structure. Such hyper-graph counting can be done for any n but quickly becomes too tedious. It is also possible to use the quantities in Table 1 to derive epidemic final sizes and other attributes.

i	1	2	3	4	5
n_i	1	3	4	4	4
l_i	0	6	8	10	12
σ_i	3	3	4	2	0
T_i	0	6	0	12	24

Table 1: Table of hyper nodes indexed by i with: the number of nodes involved n_i , the number of local links l_i , the number of global stubs σ_i , and the number of triangles T_i .

3.5.2 Motif counting algorithm

Below I introduce some notation in order to describe correctly and un-ambiguously the counting algorithm.

Path A path P is an ordered tuple $(i, \ldots, j)$.

P_n The n^{th} node of path P.

H^n The head operator such that $H^n(P)$ returns the n first nodes of the path P.

T^n The tail operator such that $T^n(P)$ returns the n last nodes of the path P.

R The reverse operator such that $R((i, \ldots, k, \ldots, j)) = (j, \ldots, k, \ldots, i)$.

$CP(P)$ The set of circular permutations of path P.

$RCP(P)$ The set of all reverse circular permutations of path P (**NB**: $RCP(P) \neq CP(P)$).

A The adjacency matrix, $A = A^T$ and with $Tr(A) = 0$.

$\{\cdot\}$ A set.

PLl The set of non-trivial paths of length l (l = number of edges).

/* *The following process is applied iteratively to determine non-trivial paths (open paths or closed paths) of length* $l+1$ *given non-trivial (non-loop) paths of length* l*. The description below is not specific to a single length but assumes* $l \geq 2$. **Data:** The uniquely counted set of paths of length 2 is: $PL_2 = \{(i,j,k)\} : A[i,k] \cdot A[k,j] > 0$ and $j > i$.

initialization;

$LL_{l+1} = \emptyset$; /* *Closed paths of length* $l+1$

$PL_{l+1} = \emptyset$; /* *Open paths of length* $l+1$

for *All paths* P *in* PL_l **do**

  for *All nodes* n*:* $A[T^1(P), n] > 0$ & $\notin T^1(P)$ **do**

    $nP = (P, n)$; /* *new path*

    if $n = H^1(L)$ & $nP \notin LL_{l+1}$ & $R(nP) \notin LL_{l+1}$ **then**

      $LL_{l+1} \leftarrow nP$;

    end

    if $n \neq H^1(L)$ & $nP \notin LL_{l+1}$ & $R(nP) \notin LL_{l+1}$ **then**

      $PL_{l+1} \leftarrow nP$;

    end

    /* *These exclude symmetric paths but not circular permutations.*

  end

  if $P^* \in \{CP(LL_{l+1}), CP(PL_{l+1})\}$ **then**

    $\{LL_{l+1}\} = \{LL_{l+1}\} \backslash P^*$;

    $\{PL_{l+1}\} = \{LL_{l+1}\} \backslash P^*$;

  end

  /* *Removes circular permutations.*

end

for *All paths* $P \in CLL_{l+1}$ **do**

  if $P^* \in CLL_{l+2}$ & $P^* \notin CRP(CLL_{l+1})$ **then**

    $LL_{l+1} \leftarrow P$;

  end

end

for *All paths* $P \in CPL_{l+1}$ **do**

  if $P^* \in CPL_{l+2}$ & $P^* \notin CRP(CPL_{l+1})$ **then**

    $PL_{l+1} \leftarrow P$;

  end

end

/* *Removes reverse circular permutations.*

Algorithm 1: Pseudo code for the motif counting algorithm.

3.5.3 Motif counting: unique vs multiplicative

In this chapter all order-four clustering-type ratios use unique counts. Ratios based on unique counts will give different values to ratios based on multiplicative counts. As an example of multiplicative counting, classic clustering is defined as:

$$\phi = \frac{6 \times [\triangle]}{[\triangle + \vee]}, \tag{3.19}$$

where $[\triangle]$ denotes the number of triangles, and $[\triangle + \vee]$ the number of length three paths closed and unclosed (doubly counted) in the network. If unique counts are used then I have $\phi_{unique} = \phi/3$.

I have computed the unique order-four counts in order to improve the computational performance of we algorithm. However, if I wish to normalise or scale-up the unique counts to correspond to the multiplicative equivalent, correct multiplying factors need to be determined. This appears to be the number of automorphisms associated with each motif type or path length: a triangle has six and a path of length three has two automorphisms.

Let $A = (a_{i,j}),\ i, j \in \{1, \ldots N\}$, be the adjacency matrix of an undirected network with no self loops i.e. $A = A^T$ and $A_{i,i} = 0$ for any $i = 1, 2, \ldots, N$. It is possible to obtain the multiplicative counts from the adjacency matrix A. Adding over all nodes:

$$[-] = \sum_{i,j=1,\ i \neq j}^{N} a_{i,j}. \tag{3.20}$$

This counts twice the number of real or uniquely counted edges in the network. It is possible to count more complex paths as well:

$$[\vee + \triangle] = \sum_{q \in Q_3}^{N} a_{i,j} a_{j,k}, \tag{3.21}$$

yielding all connected structures of 3 nodes (closed and unclosed), similarly

$$[\triangle] = \sum_{q \in Q_3}^{N} a_{i,j} a_{i,k} a_{j,k}, \tag{3.22}$$

where Q_3 denotes the set of all distinct triples of nodes yielding six times the number of unique triangles. It is also possible to count six different closed paths of length three contained within a triangle: for each node I count clockwise and counter-clockwise about the triangle. This by-directional counting is important so that the method is consistent when considering directed networks. Following the same counting methodology it is possible to count order-four structures:

$$[\sqcup + \square + \boxslash + \boxtimes] = \sum_{q \in Q_4} a_{ij} a_{jk} a_{kl}, \tag{3.23}$$

where Q_4 denotes the set of all distinct quadruples of nodes. In this form I see that it is possible to compute the individual counts using the following identities:

$$[\boxtimes] = \sum_{q \in Q_4} a_{ij} a_{ik} a_{il} a_{jk} a_{jl} a_{kl}, \tag{3.24}$$

$$[\boxslash] = \sum_{q \in Q_4} a_{ij} a_{ik} (1 - a_{il}) a_{jk} a_{jl} a_{kl}, \tag{3.25}$$

$$[\square] = \sum_{q \in Q_4} a_{ij} a_{ik} (1 - a_{il})(1 - a_{jk}) a_{jl} a_{kl}, \tag{3.26}$$

Counting this way a single $[\boxtimes]$ is counted 24 times, $[\boxslash]$ is counted 4 times and $[\square]$ is counted 8 times, equal to the number of automorphisms associated with each motif type. Currently, based on we intuition and numerical tests, I conjecture that this is the correct way to scale-up from unique to multiplicative motif counts. This method of counting is thorough but not practical for networks of reasonable size since it has complexity $\mathcal{O}(N^n)$ for order-n structures.

Chapter 4

Paper II: Beyond clustering: Mean-field dynamics on networks with arbitrary subgraph composition

Martin Ritchie [1], Luc Berthouze[2,3] & Istvan Z. Kiss[1]

[1]School of Mathematical and Physical Sciences,
Department of Mathematics, University of Sussex,
Falmer, Brighton BN1 9QH, UK.
[2] Centre for Computational Neuroscience and Robotics,
University of Sussex, Falmer, Brighton BN1 9QH, UK.
[3] Institute of Child Health, London,
University College London, London WC1E 6BT, UK
Journal of Mathematical Biology - 2015

4.1 Introduction

In this chapter, I provide a general and automated approach to deriving a set of ordinary differential equations (ODEs) that describe, to a high degree of accuracy, the expected values of prevalence or number of recovered individuals for networks that are generated based on an arbitrary set of subgraphs. This is achieved by a rigorous separation of the role of nodes within the subgraphs and by using the probability generating function (PGF) formalism to correctly track: (a) the distribution of subgraphs to which nodes belong and (b) the excess degree that is generalised from the classical notion of a stub of a single edge to different corner types given by subgraphs. This is a significant step forward as it allows us to: (a) accurately model and analyse dynamical processes on networks with higher-order structure, thus increasing model realism, (b) map out the impact of clustering in the classic sense, and more importantly, its impact at a higher level involving four or more nodes, see Chapter. 3, and (c) provide much needed insights into the role of small subgraphs or network motifs/units in epidemiology and systems biology.

The chapter is organised as follows. I first review how the probability generating function (PGF) can be used to derive ODEs that capture epidemic dynamics on configuration model (CM) networks. Such PGF-based models operate by using the versatile properties of the PGF whereby it allows us to keep track of the fraction of susceptible individuals, their degree and excess degree. Next, I generalise the CM to the *hyperstub configuration model* (HCM). The HCM is a network construction algorithm that allocates hyperstubs to nodes following a given distribution or sequence. The algorithm then selects and connects tuples of hyperstubs as prescribed by the building blocks or subgraphs of the network, rather than at random. Compared against the clustered configuration model of Chapter. 3 this is significantly more sophisticated in allowing for distributions of arbitrary subgraphs.

With a basic understanding of both the network and epidemic models, I then generalise the PGF formalism to HCM networks. This section includes a step-by-step explanation of the model derivation with examples for a particular network and a detailed presentation of the code-generating algorithm. A key component of the generalised

model is to label and track the position of each and every node in all subgraphs in order to avoid any ambiguity as to the role of nodes in non-fully-connected subgraphs. I then compare my approach to state-of-the-art models that can, in principle, capture the system's expected behaviour. Where fair comparisons are possible I show that my model displays excellent agreement with existing models, otherwise I show my model to either outperform existing models or to produce accurate results where other models fail. Finally, I use the generalised model to investigate the effect of cycles as well as the impact of higher-order structure, where global clustering is kept constant, on epidemic dynamics.

4.2 Materials and Methods

In this section I consolidate and generalise existing work centred around deriving low dimensional, deterministic and approximate ODEs that capture the time evolution of epidemic dynamics on configuration model networks. First, I re-introduce the basic susceptible-infected-recovered (SIR) epidemic model on random graphs following Volz's original PGF-based derivation [75]. This is followed by a rigorous formalisation of the hyperstub configuration model that was first presented by Karrer & Newman [32]. I then demonstrate how this model may be used to generate networks of differing subgraph compositions whilst keeping traditional network metrics such as first and second moments of the degree distribution, clustering and where possible the entire degree distribution, equal. Sec. 4.2.3 provides a derivation of the PGF-based approximate ODE model that accurately captures SIR dynamics on hyperstub configuration networks. This derivation is similar to Volz et al.'s PGF-based extension from configuration and unclustered to clustered networks [76], but generalised to incorporate arbitrary subgraphs. Finally, sec. 4.3 provides an algorithm that automatically generates and solves ODEs presented in sec. 4.2.3 for SIR epidemics on networks constructed using a user-specified set of subgraphs.

4.2.1 SIR epidemics on random graphs

The SIR compartmental model involves a population with three types of individuals – susceptible, infected or recovered – whose interactions are modelled by a network. Infection travels across edges at a per-edge rate of τ and individuals recover, independently, at rate γ. To account for the heterogeneous contact patterns, the model is centred around the PGF induced by the network's degree distribution,

$$\psi(x) = \sum_{k=0}^{\infty} p(k)x^k,$$

where $p(k)$ is the probability that a randomly chosen node has k links.

Before I can demonstrate the usefulness of storing the network in this compact way, I need to discuss the *survivor function*, $\theta(t)$, as originally presented by Volz in [75]. First, define *infectious contact* to be the event whereby an infected node v transmits to its neighbour u, regardless of the state of u, i.e., u may receive and infectious contact if it is infected or recovered, in addition to being susceptible. Next, consider an edge that has been selected uniformly at random, label its nodes u and v, and define a direction from node v to node u. The survivor function, denoted $\theta(t)$, is the probability that there has never been infectious contact from node v to node u by time t. It is at this point in the derivation that Volz proposes the following simplifying assumption, "disallow infectious contact from node u to node v". Otherwise, u may be infected by some other source, and in turn, infect v, thus increasing the probability of infectious contact from v to u, so transmission along different edges to the same target would not be independent. Disallowing infection from the test node will not affect that probability that it becomes infected and this may be used to calculate the probability that the node is still susceptible, that in turn, can be used to calculated the size of the outbreak [46]. A formal exploration of this assumption requires strong probabilistic arguments and is beyond the scope of this thesis, nonetheless Janson *et al.* and Decreusefond *et al.* have analytically shown its validity in [13, 31].

For example the probability that a degree two node is susceptible at time t is given

by $\theta(t)^2$, or more generally

$$\psi(\theta(t)) = \sum_{k=0}^{\infty} p(k)\theta(t)^k =: S(t),$$

where $S(t)$ is the fraction of susceptibles at time t. To analytically describe $\theta(t)$, I need to consider the rate at which a node with degree one becomes infected. This yields

$$\frac{d}{dt}\left(1-\theta(t)\right) = \tau\theta(t)\frac{M_{SI}(t)}{M_S(t)} \Rightarrow \frac{d\theta(t)}{dt} = -\tau\theta(t)\frac{M_{SI}(t)}{M_S(t)},$$

where $M_S(t)$ and $M_{SI}(t)$ denote the expected degree of a susceptible node and the expected number of SI edges per node at time t. Hence, $M_{SI}(t)/M_S(t)$ denotes the probability that a susceptible and infected node are connected at time t. In other words, a node which up to time t is susceptible will, on average, become infected at rate $\tau M_{SI}(t)/M_S(t)$. It turns out that $M_S(t)$ can be computed using the PGF and is given by

$$\theta(t)\left.\frac{d\psi(x)}{dx}\right|_{\theta(t)} = \sum_{k=0}^{\infty} kp(k)\theta(t)^k,$$

which can be interpreted as the expected degree conditional on nodes being susceptible. To compute $M_{SI}(t)$ additional information from the PGF must be extracted, namely the *excess degree*. This involves selecting an edge at random and following it to its originating node. The observed degree of this node, excluding the edge by which it was selected, is known as the *excess degree* and has a distribution that is generated by

$$g(z) = \frac{1}{\langle k\rangle}\sum_{k=0}^{\infty}(k+1)p(k+1)z^k.$$

As before it is possible to condition this on susceptible nodes and thus to compute the expected excess degree of susceptible nodes

$$\theta(t)\left.\frac{dg(z)}{dz}\right|_{\theta(t)} = \frac{1}{\langle k\rangle}\sum_{k=0}^{\infty} k(k+1)p(k+1)\theta(t)^k =: \delta_S(t).$$

By assuming that the expected degree of a newly infected node is equal to the expected degree of a susceptible node, Volz uses the above, multiplied by τ, to model the expected

number of edges the disease can spread across upon infection of a susceptible node. This can be used to derive the equations that describe the flux between edges in different states. Namely, these are given by

$$\begin{aligned}\frac{dM_{SS}(t)}{dt} &= -2\delta_S M_{SS}(t),\\ \frac{dM_{SI}(t)}{dt} &= -M_{SI}(t)(\tau+\gamma) + 2\delta_S(t)M_{SS}(t) - \delta_S(t)M_{SI}(t),\end{aligned}$$

where $M_{SI}(t)(\tau+\gamma)$, $2\delta_S(t)M_{SS}(t)$ and $\delta_S(t)M_{SI}(t)$ denote the I infecting the S or the I recovering, M_{SI} being created by a node in a SS edge being infected by an external source to that SS edge and, finally, the susceptible in a SI edge being infected by an external source, respectively. Summarising all the above yields the complete system of equations,

$$\begin{aligned}\frac{dS(t)}{dt} &= \frac{d\theta(t)}{dt}\psi(\theta(t)),\\ \frac{dI(t)}{dt} &= -\frac{d\theta(t)}{dt}\psi(\theta(t)) - \gamma I(t),\\ \frac{dM_{SS}(t)}{dt} &= -2\delta_S(t)M_{SS}(t),\\ \frac{dM_{SI}(t)}{dt} &= -M_{SI}(t)(\tau+\gamma) + 2\delta_S M_{SS}(t) - \delta_S(t)M_{SI}(t),\\ \frac{d\theta(t)}{dt} &= -\tau\theta(t)\frac{M_{SI}(t)}{M_S(t)},\\ R(t) &= 1 - S(t) - I(t).\end{aligned}$$

This concludes the derivation for PGF-based epidemic dynamics on random networks. Volz et al. extended this methodology to clustered networks by defining a joint probability distribution which describes the typical number of lines and triangles allocated to nodes [76]. This particular derivation has been omitted from this chapter. However, in the following section, I will outline a further generalisation of this whereby the joint probability specifies the distribution of subgraphs of various types around nodes. This then leads to more complex PGFs. In App. 4.7.3, I show how the PGF used in the main result of this chapter can be made equivalent to the PGF resulting from Volz et al.'s original edge-triangle model.

4.2.2 Hyperstub configuration model

In this chapter I generalise the configuration model [10] to the *hyperstub configuration model*. Before I specify the model I need to establish how to classify hyperstubs, the set of stubs that connect a node to a subgraph, depending on their parent subgraph and their role within that subgraph.

To generate a hyperstub configuration network model one needs to first decide on a set of subgraphs or building blocks that will form the network. This is then followed by the identification of the number of different hyperstubs induced by the subgraphs: hyperstubs must be uniquely associated with both their parent subgraph and the *orbit* of their incident nodes [32].

Definition 10. *The* orbit *of a node is the set nodes with which it may be permuted such that no edges are created or destroyed.*

For example, in Fig. 4.1, subgraph $G_{\boxslash}$ contains two distinct orbits $\{x_{14}, x_{17}\}$ and $\{x_{15}, x_{16}\}$.

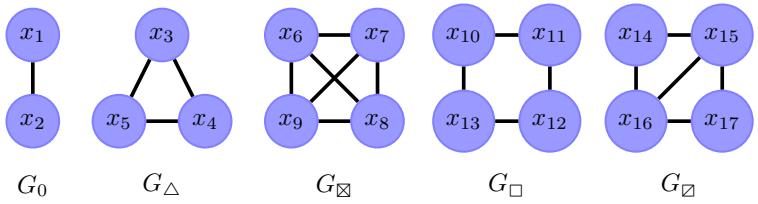


Figure 4.1: Subgraph notation and position labelling. Subgraphs are labelled by G followed by a symbolic subscript for ease of reference.

Once all hyperstubs have been identified it is possible to define a joint probability distribution that specifies the probability of a node having a certain combination of these. For example $f(x, y) = p_{x,y}$ may denote the probability of a node having $x \times G_0$ and $y \times G_\triangle$. Using this distribution it is possible to generate hyperstub degree sequences. For network generation these sequences will be subject to cardinality constraints. For example, the sum of the degree sequence of $G_\triangle$ must be divisible by three. Otherwise, the sequence needs to be re-generated. For asymmetric subgraphs, e.g., $G_{\boxslash}$, the sum of the degree sequences of both types of hyperedge must also be equal. In practice, this

can be achieved by generating a suitable degree sequence for one type of hyperedge and then randomly permute it to obtain a second sequence for the second hyperstub. $G_{\boxslash}$ has two degree sequences, one for each hyperstub, and both must be even since I select pairs of nodes from each to form the subgraph.

The network generating algorithm will then form a dynamic list for each hyperstub, where a node with hyperstub degree k_i appears k_i times. This is followed by selecting nodes from the lists, at random and without replacement, and by following the subgraphs' hyperstub composition in order to construct subgraphs and the network. It is possible that self or multi-edges form in which case the selection is discarded and new samples chosen until a valid selection is obtained. This is repeated until all lists are empty.

In this chapter I wish to both computationally generate networks and theoretically describe dynamics on such networks. The PGF of the hyperstub degree distribution provides the link between theory and simulation. The construction of the PGF induced by the hyperstub distribution can be achieved by encoding different levels of detail. At the simplest level nodes may belong to a number of subgraphs without further specifying their orbit or position within the subgraph [76]. The PGF could be constructed at the level of hyperstubs but would not differentiate between topologically equivalent positions in the subgraph, and this is what I use in my network generating algorithm (nodes may now be allocated asymmetric subgraphs) [32]. Finally, the PGF can be specified by accounting for all details described above with the addition of the precise position of nodes within the subgraph (used in the ODE derivation, sec. 4.2.3). For network generation the PGF takes the general form,

$$\psi(\hat{z}) \;=\; \sum_{\hat{h}=0}^{\infty} p_{\hat{h}} \prod_{i=1}^{m} z_i^{h_i},$$

where $\hat{z} = (z_1, z_2, \ldots, z_m)$ is a placeholder and $\hat{h} = (h_1, h_2, \ldots, h_m)$ denotes the number of h_i hyperstubs assigned to a node. The symbolic form of the PGF provides more flexibility for computation. Let us consider subgraphs distributed as follows: $G_0 \sim Pois(\lambda_1)$, $G_{\triangle} \sim Pois(\lambda_2)$ and $G_{\boxslash} \sim Pois(\lambda_3)$ (both hyperstubs of $G_{\boxslash}$ are Poisson

distributed with parameter λ_3). The PGF of such a network is

$$\psi(z_1, z_2, z_3) = \exp\left(\lambda_1(z_1 - 1) + \lambda_2(z_2 - 1) + \lambda_3(z_3 - 1)\right).$$

From this PGF, the average number of subgraphs a node belongs to may be computed

$$\left.\frac{\partial\psi(\hat{z})}{\partial z_1}\right|_{\hat{z}=1} = \lambda_1 =: \langle G_0 \rangle.$$

By replacing z_i with z^a, where a is the number of stubs contained within the hyperstub h_i, the PGF of the classical degree distribution can be recovered

$$\psi(z) := \exp\left((\lambda_1(z - 1) + \lambda_2(z^2 - 1) + \lambda_3(z^{5/2} - 1)\right).$$

The $z^{5/2}$ term accounts for the fact that $G_{\boxslash}$ is counted twice, once for each of its hyperstubs. The first and second moments of the degree distribution are directly computed using the linearity of expectation and the fact that $Var(aX) = a^2X$. As well as recovering the degree distribution, it is possible to determine the expected number of triangles per node: $\langle \triangle \rangle = \lambda_2 + 3/2\lambda_3$, since on average each node in $G_{\boxslash}$ is incident to $3/2$ triangles. To summarise, I have

$$\begin{aligned} \langle k \rangle &= \lambda_1 + 2\lambda_2 + \frac{5}{2}\lambda_3, \\ Var(k) &= \lambda_1 + 4\lambda_2 + 25/4\lambda_3, \\ \langle \triangle \rangle &= \lambda_2 + 3/2\lambda_3. \end{aligned} \tag{4.1}$$

By including a fourth subgraph in the above example, the equivalent of system Eq. (4.1) will be underdetermined with 3 equations and 4 unknowns. This allows the first and second moments and the expected number of triangles (and therefore clustering) to be fixed whilst varying the subgraph composition. For example, fixing $\langle k \rangle = 4$, $Var(k) = 8$ and $\langle \triangle \rangle = 2$, I can form the underdetermined system,

$$\begin{pmatrix} 1 & 2 & 2 & 5 \\ 1 & 4 & 4 & 25 \\ 0 & 1 & 0 & 10 \end{pmatrix} \begin{pmatrix} G_0 \\ G_{\triangle} \\ G_{\square} \\ G_{6c} \end{pmatrix} = \begin{pmatrix} 4 \\ 8 \\ 2 \end{pmatrix},$$

where the columns of the LHS matrix correspond to contributions to $\langle k \rangle$, $Var(k)$ and $\langle \triangle \rangle$ respectively and G_{ic} denotes a complete subgraph of i nodes. From this system it is possible to obtain two valid solutions: (1) $G_{\triangle} \sim Pois(2)$ and (2) $G_0 \sim Pois(9/2)$, $G_{6c} \sim Pois(3/10)$. Moreover, by replacing G_{6c} with other types of subgraph and updating the L.H.S matrix, several differing network models with the same first and second moments and clustering may be obtained. A selection of such networks used in the results section is listed below:

$$\begin{aligned}
\text{Model 1}: &\quad G_{\triangle} \sim Pois(2), \\
\text{Model 2}: &\quad G_0 \sim Pois(2),\ G_{\boxtimes} \sim Pois(2/3), \\
\text{Model 3}: &\quad G_0 \sim Pois(8/3),\ G_{5c} \sim Pois(1/3), \\
\text{Model 4}: &\quad G_0 \sim Pois(3),\ G_{6c} \sim Pois(1/5).
\end{aligned}$$

While the most basic network metrics: the average degree and global clustering coefficient, are identical their degree distributions are not. However, it is also possible to generate classes of networks where the degree distribution is equal between networks but the subgraph composition is not. Let us consider networks constructed purely out of cycles, where, regardless of the length of the cycle, cycle hyperstubs are composed of only pairs of stubs. It is then possible to increase the size of cycles whilst maintaining identical classical degree distributions between different networks. This is implemented in the following way: first, allocate to each node, on average, a pair of cycle hyperstubs, then for each type of network allow the hyperstubs to form increasingly large cycles, starting with $G_{\triangle}$ then $G_{\square}$ and so on. If the hyperstubs are distributed such that $h_i \sim Pois(2)$ then the classical degree distribution for each network will be such that only even degrees are possible, i.e., $P(degree = 2k) = P(degree = k|Pois(2))$ denoted $G_0 \sim 2Pois(2)$ for convenience. It is also possible to include a null, random, model for comparison, i.e., a network with degree distribution given by $G_0 \sim 2Pois(2)$ but connected at random. In my investigation I shall be using the following cycle-based networks:

$$\begin{aligned}
\text{Null Model}: &\quad G_0 \sim 2Pois(2), \\
\text{Model C1}: &\quad G_{\triangle} \sim Pois(2),
\end{aligned}$$

$$\text{Model C2}: \quad G_{\square} \sim Pois(2),$$
$$\text{Model C3}: \quad G_{\pentagon} \sim Pois(2),$$
$$\text{Model C4}: \quad G_{\hexagon} \sim Pois(2),$$

where $G_{\pentagon}$ and $G_{\hexagon}$ denote cycles of 5 and 6 nodes (pentagons and hexagons), respectively. Having thus created two classes of networks, the former will be used to show how conventional network metrics may not entirely capture the structure of the network as far as dynamics are concerned; the latter to investigate the effect of cycles of increasing length on dynamics.

4.2.3 SIR epidemics on hyperstub configuration model networks

This section presents the derivation of a general SIR epidemic model for a network built from an arbitrary number of subgraph types. Conceptually, this model uses the node labelling approach of [32] and generalises the PGF-type framework of Volz et al. [75, 76]. By taking this approach it is possible to derive ODEs that accurately predict the epidemic prevalence on networks that exhibit a variety of exotic subgraphs, both fully- and non-fully connected.

The first step is to choose the set of subgraphs to be included in the network. Let an arbitrary set of subgraphs be labelled by $\{G_1, G_2, \ldots, G_M\}$. For example, Fig. 4.1 shows $M = 5$ different subgraphs, which result in $m = 17$ distinct node positions, where m stands for the total number of nodes over all subgraphs. For clarity, I recall that a hyperstub is the set of half-links connecting a node to a subgraph. This example highlights the key component of the model, namely to distinguish between all nodes of a subgraph even those that are topologically equivalent. This distinction makes it possible to deal with the added complexity of having to account for labelled subgraphs. Each node/position of a subgraph is labelled. This is reflected in a PGF that accounts for each and every node in each and every subgraph. This gives rise to a PGF of the

following form:

$$\psi(\hat{\alpha}) \;=\; \sum_{\hat{y}=0}^{\infty} p_{\hat{y}} \prod_{i=1}^{m} \alpha_i^{y_i},$$

where $\hat{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m)$ is a placeholder and $\hat{y} = (y_1, y_2, \ldots, y_m)$ is such that y_i is the number of times a node appears in position x_i, $i = 1, \ldots, m$.

For each subgraph its state at time t is denoted by $G_x(S, I, \ldots, R)$. This not only describes a subgraph and its state but also the *expected* number of the given subgraph in the given state at time t, i.e., when appended with a state this notation has numerical meaning. Since $G_x(S, I, \ldots, R)$ accounts for the state of node, it will always explicitly depends on t. To describe the flux between different subgraph states, infectious events within and *between* subgraphs need to be considered. This requires a generalisation of $\theta(t)$ which was first given in sec. 4.2.1. Accordingly, I now first select a hyperstub at random and then define a direction, from its parent subgraph to its incident node. An *infectious contact* is now the event that u, regardless of its state, becomes infected by one of its adjacent nodes within that subgraph. $\theta(t)$ now needs to reflect a node's position in the subgraph. Hence, I define define $\theta_i(t)$ to be the probability that the group of edges connecting a node u in position x_i to the parent subgraph have not allowed for *infectious contact* from any infectious node in the subgraph to u by time t. Again, I impose that u cannot transmit infection to the subgraph in question. Under these assumptions, the infectious contact through hyperstubs to position x_i is now independent. A node that appears only k times in position x_i remains susceptible with probability $\theta_i^k(t)$. By geometrically compounding all $\theta_i(t)$ into a PGF, it is possible compute the fraction of the susceptible population. This is given by

$$S(t) = \psi(\hat{\theta}(t)) = \sum_{\hat{y}=0}^{\infty} p_{\hat{y}} \prod_{i=1}^{m} \theta_i(t)^{y_i}. \tag{4.2}$$

This probability is equal to the fraction of susceptible nodes in the population at time t [75]. $\theta(t)$ is referred to as a survivor function. It is dependant on time and may by computed from first principles using the definition of the Poisson process. However, in my formulation, it is computed from variables that denote the expected rate, T_i, at which infection is transmitted to a node in position x_i through the corresponding

subgraph. I note that while T is commonly used to denote the cumulative probability that infection may occur, I keep it as defined above to be consistent with the current literature on such models [76]. Each position label x_i has a T_i variable associated with it. The following examples show these rates for positions x_1, x_2 and x_3, see Fig. 4.1:

$$\begin{aligned} T_1 &= \tau[G_0(SI)], && (4.3)\\ T_2 &= \tau[G_0(IS)], && (4.4)\\ T_3 &= \tau[G_\triangle(SSI) + G_\triangle(SIS) + 2G_\triangle(SII) \\ &\quad +G_\triangle(SRI) + G_\triangle(SIR)]. && (4.5) \end{aligned}$$

To generate the above identities, I consider a susceptible node in position x_i and list all possible corresponding subgraph states that allow this node to be exposed to infection. $T = (T_1, T_2, \ldots, T_m)$ can now be used to determine the probability that a susceptible node has an infectious neighbour within a certain subgraph type. This is done by dividing $T_i\tau^{-1}$ by the number of states that involve a susceptible at position x_i:

$$\frac{T_i}{\tau \sum\limits_{A,B,C,D} G_{(\cdot)}(x_i = S, \ldots, A, B, C, D)}.$$

The expected degree of a susceptible node at position x_i is given by

$$\langle k_i \rangle = \sum_{\hat{y}=0}^{\infty} y_i p_{\hat{y}} \prod_{i=1}^{m} \theta_i^{y_i} = \theta_i \left. \frac{\partial \psi}{\partial \alpha_i} \right|_{\alpha=\hat{\theta}},$$

where $\hat{\theta} = (\theta_1, \theta_2, ..., \theta_m)$. To compute the expected degree for every position of every subgraph, one can take the Jacobian of ψ evaluated at $x = \hat{\theta}$,

$$J(\psi)|_{\alpha=\hat{\theta}}.$$

The i^{th} entry of this vector evaluated at $\alpha = \hat{\theta}$ shall be denoted J_i. A susceptible node in position x_i will have remained susceptible up to time t, with probability θ_i after which infection may be transmitted at rate T_i/J_i. This information may be used to form the following equation:

$$\frac{d}{dt}(1 - \theta_i(t)) = \theta_i(t)\frac{T_i}{J_i} \Rightarrow \frac{d\theta_i(t)}{dt} = -\theta_i(t)\frac{T_i}{J_i}. \quad (4.6)$$

$\dot{\theta}_i(t)$ decays at the rate at which a subgraph transmits infection to its node in position x_i, conditional on that node being susceptible.

Once a node is newly infected it is important to determine what, if any, subgraph states are created or destroyed. To do this, I use the susceptible nodes's excess degree prior to the infection. For the full derivation of susceptibles' excess degree refer to App. 4.7.1. In this derivation, the excess degree must be generalised to account for the degree of the different positions a node may be in, i.e., $\langle k_i \rangle,\ i = 1, 2, \ldots, m$. The expected excess degree for susceptible nodes is given by

$$\Delta_{i,j} = \theta_j \left. \frac{H_{i,j}(\psi)}{J_i(\psi)} \right|_{\alpha=\hat{\theta}},$$

where $H(\psi)$ is the Hessian of the PGF. $\Delta_{i,j}$ denotes the expected number of x_j positions associated with a node that has been selected at random, but proportionally to the number of x_i positions associated with that node. It is now possible to formulate ODEs describing the evolution subgraph states. I denote the time derivative of a subgraph's state by $\dot{G}_{(\cdot)}$. This quantity is dimensionless but not normalised. For example, the number of unique (SI) links in a network of size N is given by $[SI] = NG_0(SI)$. To form the ODE for the subgraph state $G_0(SI)$, I consider all possible ways in which this state may be created or destroyed, namely

$$\begin{aligned} \dot{G}_0(SI) &= -(\tau+\gamma)G_0(SI) \\ &\quad -(T\Delta)_1 G_0(SI) + (T\Delta)_2 G_0(SS), \end{aligned} \tag{4.7}$$

where $(T\Delta)_1$ denotes the first entry of the vector that is the product of the matrix Δ multiplied from the left by vector T. Conceptually $(T\Delta)_i$ denotes the expected number of nodes in position x_i an infection will encounter upon infecting a susceptible node through any possible route, see Fig. 4.2. The first term on the RHS of Eq. (4.7) describes this state being destroyed by the I infecting the S or the I recovering. The second term stands for this state being destroyed by the S being infected by an outside source. Finally, the last term corresponds to this state being created by the second node of $G_0(SS)$ being infected by a source external to the subgraph. To further illustrate this, the equations for $G_0(SS)$ and $G_0(IS)$ are given,

$$\dot{G}_0(SS) = -[(T\Delta)_2 + (T\Delta)_1]G_0(SS),$$

$$\begin{aligned}\dot{G}_0(IS) &= -(\tau+\gamma)G_0(II) \\ &\quad -(T\Delta)_2 G_0(IS) + (T\Delta)_1 G_0(SS).\end{aligned}$$

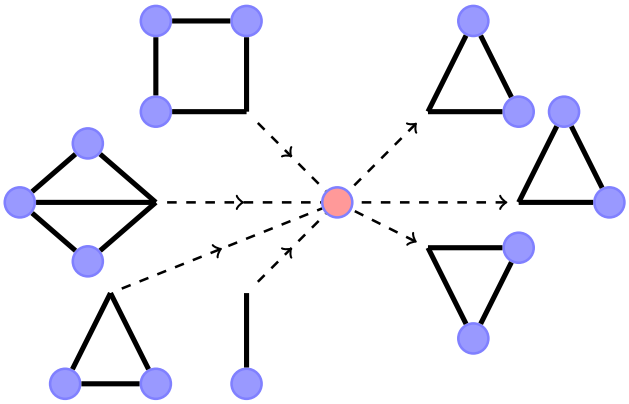


Figure 4.2: Graphical representation of $(T\Delta)_i$. Δ and T denote the excess degree of a susceptible node and rate of infection, respectively. I note that newly infected nodes are modelled as previously susceptible nodes so the product $(T\Delta)_i$ is being used to model the expected number of x_i edges infection will be able to spread along upon infecting a susceptible node. This product implicitly considers all possible routes of infection into the node. The left hand side of the figure shows example subgraphs that are the source of infection for the red node. The right hand side of the figure graphically represents the expected excess degree of $G_\triangle$ subgraphs for the red node.

Equations for every state of every subgraph must be derived. In general, I first describe any infection and recovery events of nodes within a subgraph. Next I list all possibilities for susceptible nodes to be infected from sources external to that subgraph using the appropriate $(T\Delta)$ terms.

To compute network-level prevalences, I recall that $S(t)$ can be computed at any time by Eq. (4.2). $\dot{I}(t)$ is computed directly by differentiating $S(t)$. Namely, since susceptibles become infected and and infected nodes recover at rate γ, I have

$$\dot{I}(t) = \sum_{i=1}^{m} \dot{\theta}_i(t) \frac{\partial \psi(t)}{\partial \theta_i} - \gamma I(t), \tag{4.8}$$

$$\dot{R}(t) \;=\; \gamma I(t). \tag{4.9}$$

The total number of equations is given by $2 + m + \sum_{i=1}^{M} 3^{|G_i|}$, where $|\cdot|$ denotes the number of nodes in a subgraph. In App. 4.7.2 I give or more example ODEs and in App. 4.7.3 I show how my model is equivalent to previous systems developed for complete subgraphs [76].

4.2.4 Initial conditions

Let ϵ be the fraction of initially infected nodes. Hence, $\epsilon = I_0/N$, where I_0 is the number of initially infected nodes and N is the network size. Initial conditions for the I and R populations are given by

$$I(0) = \epsilon,\ R(0) = 0.$$

At time $t = 0$ no hyperstub has transmitted infection, therefore, $\theta_i(t = 0) = 1$. For a subgraph that contains a single infected node, $G(t = 0) = \epsilon\langle k\rangle_i$ where $\langle k\rangle_i$ is the expected hyperstub degree. For the subgraph with every node susceptible I set $G(t = 0) = (1-\epsilon)\langle k\rangle_i$. By assuming that only a small fraction of the population, i.e., a single node, is initially infected, I do not allow non-zero initial conditions for subgraphs with more than one infectious node.

4.3 Automated code-generation of the mean-field model

I now present my methodology for computationally generating a complete system of equations for a network constructed from subgraphs following a configuration model. This procedure requires the PGF of a hyperstub degree distribution (HDD), the adjacency matrices of corresponding subgraphs, and epidemiological parameters as inputs. The algorithm will output the system of ODEs that will predict the network-level prevalence. Table 4.1 gives a brief summary of the variables that need to be generated, listed in the order they are generated in this section.

Variable	Description	Generation
ψ	PGF of the HDD given as a function, not as a series.	A symbolic software package can be used to compute the Jacobian and Hessian.
$\theta_i(t)$	Survivor functions with their evolution equations given by ODEs.	These ODEs can be defined within a single `for` loop, see Eq.(4.6).
(S, I, R)	The prevalences of S, I and R, with the latter two given by numerical solutions of ODEs	From Eq. (4.8), it follows that $S = \psi(\theta)$.
T_i	Total rate of infection experienced by an S in position x_i.	For a subgraph with m nodes, T_i may be generated by m nested `for` loops cycling through the possible states that a subgraph can be in, see Eq. (4.3).
$G_x(S, I, \ldots, R)$	Expected prevalence of a subgraph in a given state.	The equation for this is computed based on the rate matrix, $\mathbf{Z}$, see Eq. (4.10).

Table 4.1: Summary of the key system variables and their generation.

Let $\vec{G}$ denote the vector of states of a subgraph G with $\vec{G}_i$ denoting a specific state of G. For the SIR model, $\vec{G}$ has $3^{|G|}$ elements. To generate T_i from $\vec{G}$, the following steps are needed: (1) cycle through $\vec{G}$, (2) for each infectious contact to node i in state $\vec{G}_j$, update T_i to $T_i = T_i + \vec{G}$. Using T the survivor functions can be computed, see Eq. (4.6), which are then used to compute the fraction of the population which is susceptible, infected or recovered, see Eq. (4.8).

The ODEs corresponding to subgraphs need to be represented with a *rate matrix*, $\mathbf{Z}$. This matrix encodes all information relating to the given subgraph, namely the excess degrees, rates of infection over subgraphs T, epidemiological parameters τ and γ, and implicitly encodes the subgraph's adjacency matrix g. To compute Δ, I use Eq. (4.2) and a symbolic software package to calculate the Jacobian and Hessian of the PGF.

For each subgraph, I initialise the matrix $\mathbf{Z}$ as a square matrix with all entries set to zero. The i^{th} column and row of $\mathbf{Z}$ correspond to state $\vec{G}_i$. Once populated, the entry $\mathbf{Z}_{i,j}$ contains the rate at which state i transitions to state j.

To illustrate how to generate $\mathbf{Z}$, I consider the G_0 subgraph, see Fig. 4.1, with states $\vec{G} = (SS,\, SI,\, SR,\, IS,\, II,\, IR,\, RS,\, RI,\, RR)$. I associate the state $G_0(SS)$ with the first row and column of $\mathbf{Z}$. Moving along the top row, when a column index is reached that corresponds to a state that $G_0(SS)$ may transition to, I update the entry with the appropriate rate. The first row of $\mathbf{Z}$ is all zero except for $\mathbf{Z}_{1,2} = (T\Delta)_2$ and $\mathbf{Z}_{1,4} = (T\Delta)_1$. The second row, corresponding to state $G_0(SI)$, has entries $\mathbf{Z}_{2,3} = \gamma$ and $\mathbf{Z}_{2,5} = \tau + (T\Delta)_1$, see Eq. (4.7). Fill every row of the matrix $\mathbf{Z}$ in this way, refer to App.4.7.4 the full matrix corresponding to G_0. The algorithm for this process is given for an arbitrary subgraph in App. 4.7.6, and the corresponding Matlab code is provided as supplemental material.

Using the rate matrix, the ODE for the subgraph state $\vec{G}_i$ yields

$$\frac{d\vec{G}_i}{dt} = -\left(\sum_{j=1}^{3^{|G|}} \mathbf{Z_{i,j}}\right)\vec{G}_i + \left(\sum_{k=1}^{3^{|G|}} \mathbf{Z_{k,i}}\right)\vec{G}_k. \tag{4.10}$$

where $|G|$ and $\vec{G}_i$ denote the number of states the subgraph G may take and a specific state of G respectively. The final step to generating the full system is to set the initial conditions. Only the initial conditions for subgraph states need computing as $I(0),\ R(0)$

and $\theta_i(0)$ are fixed as per the previous section. This can be done by cycling through each element of $\vec{G}$. If (a) $\vec{G}_i$ is a purely susceptible state then I set $\vec{G}_{i_0} = J_i(1-\epsilon)$, and if (b) $\vec{G}_i$ contains a single infectious individual and is otherwise susceptible, I set $\vec{G}_{i_0} = J_i\epsilon$. All other states are set to zero, as I assume that with a sufficiently small infectious seed, the probability of having two infectious individuals in a subgraph is zero.

4.4 Results

To validate the proposed mean-field model and to assess the goodness of the approximation, I compare results from the ODEs to output from stochastic simulations. Networks were generated following the configuration algorithm, please refer to App. 4.7.5. Typically I generated 500 networks of size $N = 15000$ and computed a single realisation of the epidemic, according to the Gillespie algorithm with the per link rate of infection $\tau = 1$ and a recovery rate of $\gamma = 1$. Simulations which died out before an outbreak occurred were removed. The simulations were seeded with a single infectious individual and an outbreak was said to occur if 5% infectious prevalence was achieved. In all plots simulation results and the solution of ODEs are plotted in solid lines and discrete points, respectively.

To start, I test the performance of my model against existing or state of the art models. To do this, in Fig. (4.3), I show results for two degree distributions that are homogeneous in the classical sense. Their PGFs are given by

$$\begin{aligned}\psi_1(\hat{\alpha}) &= \frac{1}{2}(\alpha_{14}+\alpha_{17})\frac{1}{2}(\alpha_{15}+\alpha_{16}),\\ \psi_2(\hat{\alpha}) &= \frac{1}{2}(\alpha_1+\alpha_2)\frac{1}{4^2}(\alpha_{10}+\alpha_{11}+\alpha_{12}+\alpha_{13})^2,\end{aligned}$$

where the variables α_i correspond to subgraphs given in Fig. 4.1. Figure 4.3 shows results from a pairwise model with closures at the level of quadruples [26, 28]. While the classical clustering is easy to compute, the order-four clustering/transitivity ratios were measured following a recently developed subgraph counting algorithm, see Chapter. 3. These are defined as the ratio of a given subgraph count to all open and closed paths of

length four, both counted uniquely. Currently, this model operates using an average or homogenous degree and stores no information about the degree distribution, but does assume random mixing of subgraphs.

All models perform well in capturing the epidemic dynamics on networks generated using the PGF given by ψ_1, see Fig. (4.3) with higher epidemic peak. However, when networks are created using the PGF given by ψ_2, see lower peak in Fig. (4.3), the pairwise model struggles to accurately capture the dynamics, both anticipating and compressing the epidemic's time scale or duration, but predicting correctly the peak prevalence. The pairwise model does not encode any information relating to degree or subgraph distribution and hence a homogeneous random set-up, as used here, would be an appropriate choice.

The key advantage of my algorithm over existing ones is that it can handle non-fully connected subgraphs. To test this, in Fig. 4.4, I utilise networks models C1-C4 as specified in sec. 4.2.2. Fig. 4.4 shows plots of simulation average compared to the ODE's solution for the four network types. I observe that the epidemic behaviour of networks composed of increasingly large cycles quickly converge to that of the random null case.

It has previously been observed that for networks with the same degree distribution, an increasing level of clustering slows the epidemic transmission and requires a higher transmission rate in order to observe a successfully spreading epidemic [20, 33]. This occurs for two reasons: (1) subgraphs that are densely connected share fewer connections to the rest of the network so an initial seed will be restricted to one part of the network and (2) this same effect leads to infectious nodes competing for susceptible nodes. While this may make transmission more efficient locally, it does limit further seeding in fully susceptible parts of the network. Fig. 4.4 shows that the effect of $G_{\square}$ is similar to that of the clustered network, but less pronounced; both the time and size of peak infectious prevalence is delayed and reduced when compared to the null case. For cycles larger than four nodes this behaviour is less pronounced and the epidemics for larger cycles converge to the null case, as observed with $G_{\pentagon}$.

To highlight the flexibility of my model and its wide-ranging applicability to systematically investigating the impact of higher-order network structure, in Fig. (4.5), I

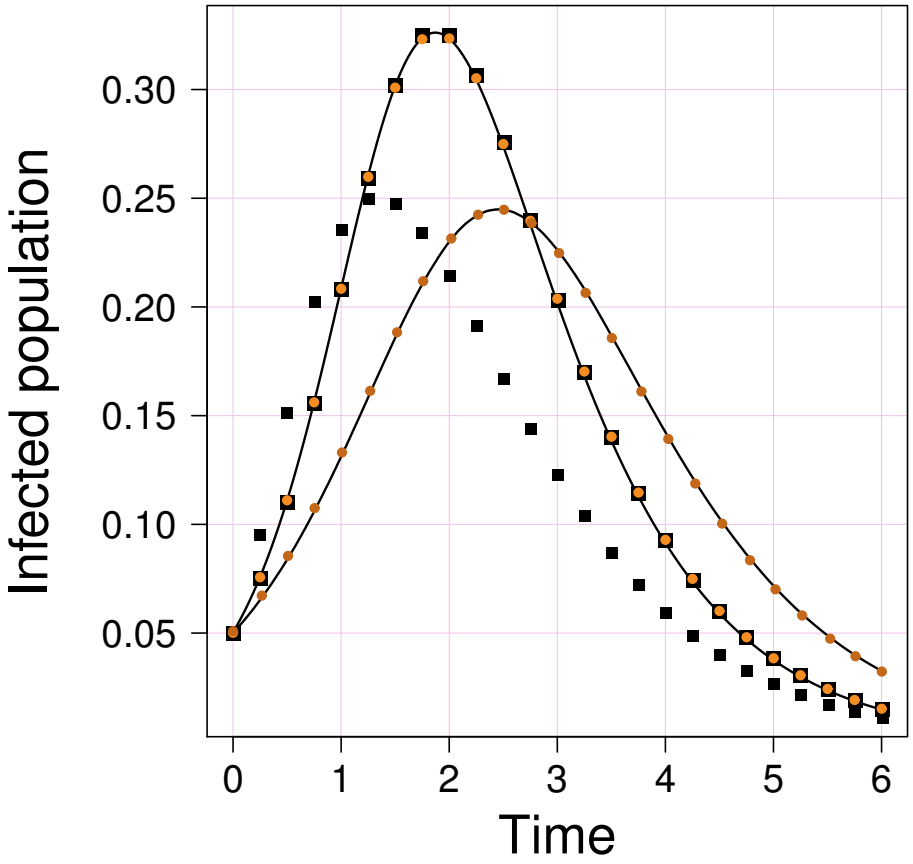


Figure 4.3: Performance of other models. Lines, circles and squares correspond to simulation average, ODE solution and pairwise ODE solution, respectively. All networks are homogeneous with $k = 5$. The lower peaks correspond to networks generated with each node allocated one of each corner type of a $G_{\boxslash}$ with clustering $\phi = 0.3$. Data with higher peak correspond to networks generated with a single G_0 and two $G_{\square}$ subgraphs yielding $\phi \approx 0$.

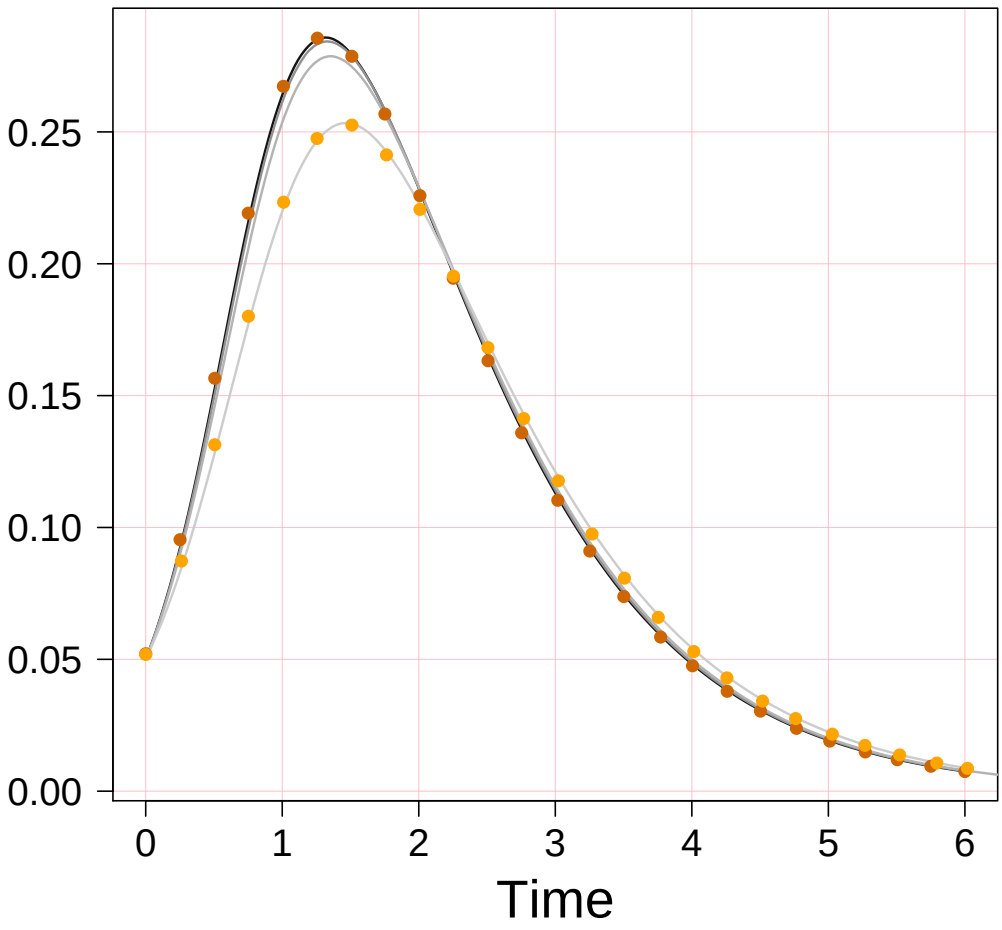


Figure 4.4: Clustering and cycles. Solid lines and markers correspond to simulation average and ODE solution, respectively. From darkest to lightest, the solid lines correspond to: $k \sim 2Pois(2)$, $G_{\pentagon} \sim Pois(2)$, $G_{\square} \sim Pois(2)$ and $G_{\triangle} \sim Pois(2)$, i.e., each network used has an identical degree distribution given by $P(degree = 2k) = P(degree = k|Pois(2))$. Clustering is $\phi = 0.2$ and $\phi \approx 0$ for the $G_{\triangle}$ and other networks, respectively. For clarity, ODE solutions for only the two extreme cases, the null and triangle network, have been included. Epidemics corresponding to cycles of length six have been computed but omitted due to their close similarity to the null case. Only two ODE solutions have been included for upper and lower cases.

consider four networks with the same first and second moments, and the same classical clustering but generated using different families of subgraphs, see models 1-4 sec. 4.2.2. Figure 4.5 shows simulation average for all four networks and the solution of ODEs for the upper and lower cases, models 1 and 4 respectively.

Figure 4.5 shows a clear trend whereby larger subgraphs lead to epidemics with smaller peak prevalence. A second more subtle trend shows a delay in time until peak prevalence. Subgraphs of larger size lead to a significant difference in the behaviour of epidemics and echo what was observed for increasing levels in clustering. This could be explained by considering a subgraph with average degree $\langle k_s \rangle$. When $\langle k \rangle < \langle k_s \rangle$ the network will exhibit extreme clustering, where isolated structures are increasingly densely connected at the cost of becoming isolated. This effect is more subtle than clustering but it can be significant. This suggests that the accuracy of future models would improve if they can correctly account for networks' subgraph composition, particularly subgraphs beyond that of triangles.

Finally, the data in Fig. 4.5 has been produced using networks that do not have the same degree distribution but do have equal first and second moments, and clustering. To better understand how the non-equal higher moments may have affected the results, I have simulated epidemics on the corresponding random networks, Model $1'$: $G_0 \sim 2Pois(2)$ and Model $4'$: $G_0 \sim Pois(3) + 5Pois(1/5)$, see App. 4.7.7. This plot shows that the differences observed in Fig. 4.5 cannot be explained by the difference in the degree distribution alone. Thus, generating identical clustering but using different subgraphs can lead to non-negligible differences in epidemic dynamics.

4.5 Discussion

Higher-order structures, captured for example as different subgraph compositions and arrangements in a network, have been identified as features of real networks. Examples include households, social interactions and biological networks. These building blocks of networks have been shown to play a key role in defining a network's topology and can have significant impact on the functions of the network or on the dynamical processes unfolding on the network. Despite this, the modelling toolset in this direction

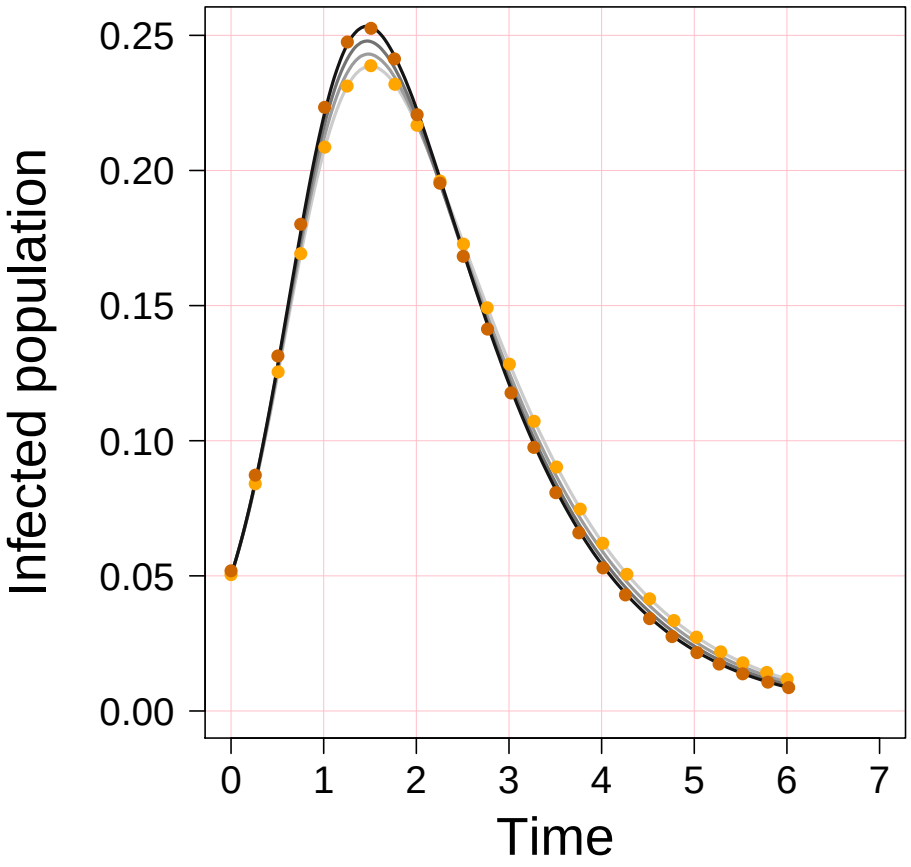


Figure 4.5: Clustering via differing subgraphs. Solid lines and markers correspond to simulation average and ODE solution, respectively. From darkest to lightest the solid lines correspond to: $G_{\triangle} \sim Pois(2)$; $G_0 \sim Pois(2)$, $G_{\boxtimes} \sim Pois(2/3)$; $G_0 \sim Pois(8/3)$, $G_{cp}, \sim Pois(1/3)$ and $G_0 \sim Pois(3)$, $G_{ch}, \sim Pois(1/5)$, where cp and ch denote complete pentagon and hexagon subgraphs, respectively. The networks were generated so that $\langle k \rangle = 4$, $var(k) = 8$ and $\phi = 0.2$. The downward trend of peak prevalence corresponds to networks composed of complete subgraphs of increasing size. The larger subgraphs lead to more connections within the group rather than to the rest of the network.

is underdeveloped. Here, I provided an approach that considerably extends the scope of the current modelling framework by enabling us to consider arbitrary sets of exotic subgraphs as building blocks for the network. My approach also offers control over the arrangements of subgraphs and, more importantly and uniquely, an automated way of generating a system of ODEs that accurately capture the prevalence profile for a wide range of subgraph sets, as shown in the results section.

The previous section has shown how higher-order structures may be investigated using this model. Moreover, I provided the first example of generating classes of networks constructed using different subgraph sets while keeping degree, variance and clustering, all in the classic sense, fixed. For example, I showed that epidemics on networks with no clustering, but exhibiting cycles, display features which are significantly different to those observed in classical random networks with effectively no clustering. Equally, I have shown that different subgraph combinations or arrangements can create higher-order structure that may significantly affect the epidemic dynamics. My work opens the possibility to carry out a wide-ranging and systematic investigation of the impact of subgraphs and higher-order structure on dynamics on networks. When presented with real world network data whose structure can be explained by a set of subgraphs, all that will be needed in order to apply my framework is to extract the subgraphs and their distribution around nodes. A possible limitation to the widest applicability is the number of nodes in the largest subgraph. However, as shown by my results when going from squares to pentagons, it is likely that the effect of higher-order structures will decay, or be less marked, as their size increases.

There are two key ways in which this work may be extended: (a) generalisation to SIS dynamics. Due to the definition of $\theta(t)$ it is currently not possible to apply this model to SIS dynamics. However, all the framework relating to network structure is independent from this variable and may therefore still be appropriate. (b) The subgraph approach is highly suitable for adaptation to household models. Household models typically specify a distribution of household sizes overlaid on a contact network to produce a well-connected network [4, 25]. A successful incorporation of such network in my framework could lead to a highly relevant set of household models.

4.6 Acknowledgements

Martin Ritchie acknowledges funding for his PhD studies from EPSRC (Engineering and Physical Sciences Research Council) and the University of Sussex.

4.7 Appendix

In this Appendix I (a) give a more detailed explanation of the excess degree, (b) provide ODEs for an example network, (c) show how my generalised model reduces to a previous model under certain conditions, (d) provide an example state transition matrix, (e) give pseudocode for both the subgraph-based configuration model and the algorithm used to generate the state transition matrix and, finally, (f) compare epidemic dynamics on two configuration model networks with their degree distributions being different but with the same mean and variance.

4.7.1 Excess degree

Recall the probability generating function (PGF) of a network's hyperstub degree distribution with m nodal positions:

$$\psi(\hat{\alpha}) \;=\; \sum_{\hat{y}=0}^{\infty} p_{\hat{y}} \prod_{i=1}^{m} \alpha_i^{y_i}, \tag{4.11}$$

where $\hat{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m)$ is a placeholder, and $\hat{y} = (y_1, y_2, \ldots, y_m)$ is such that y_i denotes the number of times a node appears in position x_i, $i = 1, 2, \ldots, m$. The PGF of the excess degree distribution is a critical component in my derivation and it is illustrative to see how it is computed. To see this, I first compute the expected excess degree, select a node at random but proportional to its number of x_i hyperstubs, $y_i p_{\hat{y}}$. Next, to obtain the expected x_i degree, sum this product over all nodes

$$\langle x_i \rangle \;=\; \sum_{\hat{y}=0}^{\infty} y_i p_{\hat{y}} =: \left. \frac{\partial \psi(\hat{\alpha})}{\partial \alpha_i} \right|_{\hat{\alpha}=\mathbf{1}}. \tag{4.12}$$

The above sum considers each and every node from which an x_i hyperstub originates. However, in hyperstub configuration model networks there is usually more than one

type of hyperstub and this adds an additional level of detail to the excess degree. The excess degree now may incorporate two different hyperstubs into its calculations. It is now possible to describe a nodes x_i degree but conditional on it being selected through ones of its x_j hyperstubs. More formally I can compute the expected excess degree using conditional expectation, $E(x_j|x_i = y_i)$, which yields

$$\delta_{x_j,x_i} = \frac{\sum_{\hat{y}=0}^{\infty} y_j y_i p_{\hat{y}}}{\sum_{\hat{y}=0}^{\infty} y_i p_{\hat{y}}}, \tag{4.13}$$

where δ_{x_j,x_i} denotes the expected x_i hyperstub degree observed from a node selected proportionally to its x_j hyperstub degree. The denominator is given by Eq. (4.11), and the numerator is specified by

$$\sum_{\hat{y}^*=0}^{\infty} y_i y_j p_{\hat{y}^*} = \left.\frac{\partial^2 \psi}{\partial \alpha_i \alpha_j}\right|_{\alpha=\mathbf{1}}.$$

4.7.2 ODEs for an example network

The following provides ODEs for a simple example network composed of only G_0 and $G_\triangle$.

When deriving ODEs by hand listing out equations for T_i is a good starting point as they include many of the subgraph states, i.e. $G_0(SI)$, and can be used as the start of a check list when listing state equations

$$\begin{aligned}
T_1 &= \tau[G_0(SI)], \\
T_2 &= \tau[G_0(IS)], \\
T_3 &= \tau[G_\triangle(SSI) + G_\triangle(SIS) + 2G_\triangle(SII) \\
&\quad + G_\triangle(SRI) + G_\triangle(SIR)], \\
T_4 &= \tau[G_\triangle(SSI) + G_\triangle(ISS) + 2G_\triangle(ISI) \\
&\quad + G_\triangle(RSI) + G_\triangle(ISR)], \\
T_5 &= \tau[G_\triangle(ISS) + G_\triangle(SIS) + 2G_\triangle(IIS) \\
&\quad + G_\triangle(IRS) + G_\triangle(RIS)].
\end{aligned}$$

It is important to node the above equations will not list every subgraph state and that for a subgraph composed of n will have 3^n state equations. For example, the first few state equation for G_0 are given by

$$\begin{aligned}
\dot{G}_0(SS) &= -[(T\Delta)_2 + (T\Delta)_1]G_0(SS), \\
\dot{G}_0(SI) &= -(\tau+\gamma)G_0(SI) - (T\Delta)_1 G_0(SI) + (T\Delta)_2 G_0(SS), \\
\dot{G}_0(IS) &= -(\tau+\gamma)G_0(IS) - (T\Delta)_2 G_0(IS) + (T\Delta)_1 G_0(SS),
\end{aligned}$$

with equations for the following being omitted

$$\{\dot{G}_0(SR),\ \dot{G}_0(II),\ \dot{G}_0(IR),\ \dot{G}_0(RS),\ \dot{G}_0(RI), \dot{G}_0(RR)\},$$

Similarly, sample ODEs for the $G_\triangle$ subgraph, taken from a system of 27 ODEs, are:

$$\begin{aligned}
\dot{G}_\triangle(SSS) &= -[(T\Delta)_5 + (T\Delta)_4 + (T\Delta)_3]G_\triangle(SSS), \\
\dot{G}_\triangle(SSI) &= -[2\tau + \gamma + (T\Delta)_4 + (T\Delta)_3]G_\triangle(SSI) \\
&\quad +(T\Delta)_5 G_\triangle(SSS), \\
\dot{G}_\triangle(SIS) &= -[2\tau + \gamma + (T\Delta)_5 + (T\Delta)_3]G_\triangle(SIS) \\
&\quad +(T\Delta)_4 G_\triangle(SSS), \\
\dot{G}_\triangle(ISS) &= -[2\tau + \gamma + (T\Delta)_5 + (T\Delta)_4]G_\triangle(ISS) \\
&\quad +(T\Delta)_3 G_\triangle(SSS),
\end{aligned}$$

with equations for the following being omitted

$$\begin{aligned}
&\{\dot{G}_0(SSR),\ \dot{G}_0(SII),\ \dot{G}_0(SIR),\ \dot{G}_0(SRS),\ \dot{G}_0(SRI),\ \dot{G}_0(SRR), \\
&\dot{G}_0(ISI),\ \dot{G}_0(ISR),\ \dot{G}_0(IIS),\ \dot{G}_0(III),\ \dot{G}_0(IIR),\ \dot{G}_0(IRS), \\
&\dot{G}_0(IRI), \dot{G}_0(IRR), \dot{G}_0(RSS),\ \dot{G}_0(RSI),\ \dot{G}_0(RSR),\ \dot{G}_0(RIS), \\
&\dot{G}_0(RII), \dot{G}_0(RIR), \dot{G}_0(RRS), \dot{G}_0(RRI),\ \dot{G}_0(RRR)\},
\end{aligned}$$

Each hyperstub will have a survivor function and a corresponding ODE describing its evolution, as follows:

$$\dot{\theta}_1 = -\theta_1 \frac{T_1}{M_1},$$

$$\begin{aligned}
\dot{\theta}_2 &= -\theta_2 \frac{T_2}{M_2}, \\
\dot{\theta}_3 &= -\theta_3 \frac{T_3}{M_3}, \\
\dot{\theta}_4 &= -\theta_4 \frac{T_4}{M_4}, \\
\dot{\theta}_5 &= -\theta_5 \frac{T_5}{M_5}.
\end{aligned}$$

The fraction of the population that is susceptible or infected is computed by compounding θ_i into the PGF. Symbolically, this is computed by the following

$$\begin{aligned}
\dot{S} &= \frac{d}{dt}\psi(\hat{\theta}), \\
\dot{I} &= -\frac{d}{dt}\psi(\hat{\theta}) - \gamma I, \\
R &= \gamma I,
\end{aligned}$$

where ψ is the probability generating function that generates the hyperstub degree distribution and $\hat{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$ is the probability that infection via subgraphs of types one to five has not been transmitted. The total system size for this example network is given by

$$3^2 + 3^3 + 5 + 2 = 43,$$

with each term in the above corresponding to G_0, $G_\triangle$, survivor functions and epidemic prevalence, respectively. In general, the total number of equations is given by:

$$\sum_{i=1}^{M} 3^{|G_i|} + |G_i| + 2,$$

where G_i denotes a subgraph, $|G_i|$ is the number of nodes in a subgraph, and m is the total number of subgraphs.

4.7.3 Equivalence to previous model for complete subgraphs

The PGF formulation originally proposed by Volz et al. [76] is equivalent to my proposed model in the case of complete subgraphs. Consider an arbitrary complete subgraph composed of l nodes and a network that is composed only of this subgraph. If

positions within the subgraph are labelled distinctly, $\{x_1, x_2, \ldots x_l\}$, as I have done in my approach, then the PGF of such a network is given by

$$\psi_p(\hat{\alpha}) = \sum_{\hat{y}=0}^{\infty} p_{\hat{y}} \prod_{i=1}^{l} \alpha_i^{y_i}, \tag{4.14}$$

where $\hat{y} = (y_1, y_2, \ldots, y_l)$. Volz et al.'s framework treats all topologically equivalent positions as one single position. Thus, in this case, the subgraph has a single label, x, that corresponds to a single count, y, and the PGF takes the following form

$$\psi_v(\hat{\alpha}) = \sum_{y=0}^{\infty} p_y \alpha^y. \tag{4.15}$$

I now show how one may obtain Eq. (4.15) from Eq. (4.14). Since both PGFs describe the same network, the rate at which my formulation allocates position x_i must be $1/l$ the rate at which Volz et al.'s formulation allocates x. If I replace the unique position labels of Eq. (4.14) with a single position marker (such as in Volz et al.'s model), the following expression is obtained

$$\psi_p(\hat{\alpha}) = \sum_{\hat{y}=0}^{\infty} p_{\hat{y}} \prod_{i=1}^{l} \alpha^{y/l}, \tag{4.16}$$

where the following substitutions, $y_i = y/l$ and $\alpha_i = \alpha$, were made so that α^y is the result of the above product. Now, every time an x_i is allocated, I allocate an x instead. Finally, since $p_{\hat{y}}$ is a joint distribution of l identically distributed independent random variables, i.e., $\hat{y} = (y/l, y/l, \ldots, y/l)$, I get:

$$\psi_p(\hat{\alpha} = \alpha) = \sum_{y=0}^{\infty} p_y \alpha^y.$$

It is also possible to translate between the two models elsewhere in the derivation. As an example, in my approach, infection over lines is given by T_1 and T_2, as per Eq.(4.3). By adding these values, the equivalent values used in Volz et al.'s formulation may be recovered. Following my derivation, first let $G_0(SI) \equiv G_0(IS)$ and:

$$T_1 + T_2 = \tau G_0(SI) + \tau G_0(IS) = 2\tau G_0(SI).$$

Since each G_0 is generated from a PGF that allocates positions at rate 1/2 that of Volz et al.'s PGF, the 2 will cancel yielding $\tau G_0(SI)$. However, it is only necessary to show equivalence between the two PGFs since all other variables follow from this.

4.7.4 State transition matrix

The state transition matrix for G_0 (lines) is given by:

$$\mathbf{Z} = \begin{array}{c} \\ (SS) \\ (SI) \\ (SR) \\ (IS) \\ (II) \\ (IR) \\ (RS) \\ (RI) \\ (RR) \end{array} \begin{array}{c} \begin{array}{ccccccccc} (SS) & (SI) & (SR) & (IS) & (II) & (IR) & (RS) & (RI) & (RR) \end{array} \\ \begin{pmatrix} 0 & (T\Delta)_2 & 0 & (T\Delta)_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \gamma & 0 & \tau + (T\Delta)_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & (T\Delta)_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \tau + (T\Delta)_2 & 0 & \gamma & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \gamma & 0 & \gamma & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & (T\Delta)_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

4.7.5 Algorithm 1 - Hyperstub CM algorithm

input : N, K,

output: A.

Variables / initialisation

N: the number of nodes,

% Each row of K corresponds to single node's hyperstub sequence.

K: the hyperstub degree sequence, a non-square matrix $K \in \mathbb{N}_0^{N \times H}$,

H: the number of hyperstub types,

A: the adjacency matrix of the network, $A \in \{0,1\}^{N \times N}$,

M: the number of subgraphs,

H_i: the degree of a specific hyperstub,

h_i: a dynamic list of nodes that are incident to H_i hyperstubs,

g_i: the adjacency matrix of a subgraph, $g \in \{0,1\}^{n_i \times n_i}$,

n_i: the number of nodes in g_i.

Procedure

% The following creates dynamic lists the, 'hyperstub bins'.

for every node i **do**

  for each H_j **do**

    append $K_{i,j}$ multiples of $node(i)$ to the hyperstub bin(h_j)

  end

end

⋮

```
⋮
for For each subgraph g_i do
    for For each hyperstub of g_i do
        % Select unfiformly at random and without
        % replacement a node incident to each desired hyperstub.
        n_1 = rand-sample(h_{i_1})
        n_2 = rand-sample(h_{i_2})
            ⋮
        n_{g_i} = rand-sample(h_{i_2})
    end
    % The following compares pairs of the selected nodes
    % to determine their connectivity in A.
    for k = (1, 2, ..., n_i) do
        for l = (1, 2, ..., n_i) do
            if g(n_k, n_l) == 1 then
                A(n_k, n_l) = 1
            end
        end
    end
end
```

Algorithm 2: The hyperstub configuration model. In this implementation, multiple-edges are over written but self-edges are permitted. To prevent this, if nodes already share an edge or a node has been selected twice (self-edge) the current selection of nodes is disregarded and a new selection is made, this is repeated until a valid selection is made. This reselection step has been omitted below for readability.

4.7.6 Algorithm 2 - Transition matrix algorithm

input : g,

output: $\mathbf{Z}$.

Variables / initialisation

g: the adjacency matrix of a subgraph G,

% $\mathbf{Z} \in \mathbb{R}^{3^n \times 3^n}$.

$\mathbf{Z}$: matrix corresponding rate of transition between states of G,

n: node count of G,

% $\vec{G}$ contains 3^n elements.

$\vec{G}$: the vector of states of G,

τ: per link infection rate,

γ: recovery rate,

$T\Delta$: the expected force of infection a node within G experiences from outside G.

$\vdots$

⋮

Procedure

for *every state* $\vec{G}_i$ **do**

  for *every state* $\vec{G}_j$ **do**

    % Compare each and every possible state transition of G:

    switch $\vec{G}_i \rightarrow \vec{G}_j$ **do**

      case *A single infection occurs* **do**

        if *the new I is connected to another I within G* **then**

          % Check the connectivity of the new I using g.

          $Z_{i,j} = \tau + T\Delta$

        else

          % the infection was from only an external source.

          $Z_{i,j} = T\Delta$

        end

      end

      case *A single recovery occurs* **do**

        $Z_{i,j} = \gamma$

      end

      case *otherwise* **do**

        $Z_{i,j} = 0$

      end

    end

  end

end

Algorithm 3: Generating the state transition matrix. The switch comparison needs to check: (1) that only a single node has changed state and (2) only state changes $S \rightarrow I$ and $I \rightarrow R$ are valid.

4.7.7 Null case for Fig. 4.5

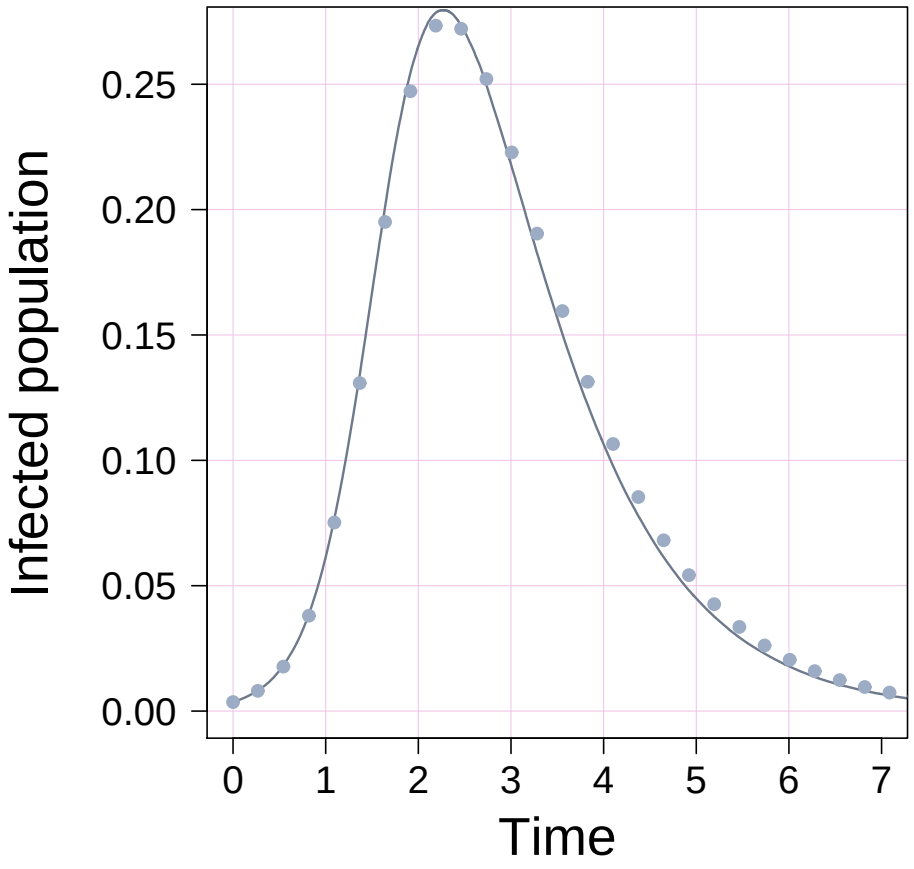


Figure 4.6: The effect of higher moments. The solid and discrete plots correspond to the null networks $G_0 \sim 2Pois(2)$ and $G_0 \sim Pois(3) + 5Pois(1/5)$ respectively, i.e. the null cases for the triangle and hexagon networks. Both plots have equal first and second moments and clustering equal to that of a random network. The difference observed is a result of non-equal higher moments and is not enough to explain the difference observed in Fig. 4.5.

Chapter 5

Paper III: Generation and analysis of networks with a prescribed degree sequence and subgraph family: Higher-order structure matters

Martin Ritchie [1], Luc Berthouze[2,3] & Istvan Z. Kiss[1]

[1]School of Mathematical and Physical Sciences,
Department of Mathematics, University of Sussex,
Falmer, Brighton BN1 9QH, UK.
[2] Centre for Computational Neuroscience and Robotics,
University of Sussex, Falmer, Brighton BN1 9QH, UK.
[3] Institute of Child Health, London,
University College London, London WC1E 6BT, UK
Journal of Complex networks - in press

5.1 Introduction

In the standard configuration model, triangle subgraphs appear infrequently as a by-product of working with finite size networks [10]. But what if one *wants* triangle subgraphs to appear in a network, in particular, if one wants to model a complex network with clustering? An extension of the configuration model to this case exists [45, 58]. In this extension a node is allocated a number of stubs, that may go on to form standard edges, as well as a number of triangle 'corners' or *hyperstubs*, pairs of stubs that will form triangles. While edges are formed in the usual way, triangles are formed by selecting three triangle hyperstubs at random and connecting their pairs of constituent stubs.

As for edges the number of all stubs must be divisible by two, the total number of triangle hyperstubs must be divisible by three is a necessary condition for the triangle hyperstub sequence to be graphical. Another similarity this model shares with the standard configuration model is that the probability that any two triangles will share an edge, thus forming a $G_{\boxslash}$ subgraph (see Figure 5.1), vanishes in the limit of large network size [32]. Just as a network composed of lines only is limited in recreating real-world networks, so is a model that can only include edges and triangles. Obviously, this may depend on properties and structure of the real networks, but in many cases edges and triangles are not enough to produce an accurate enough artificial replica of the real network.

The configuration model has since received further attention to address this [32]. Building on the edge-triangle model a more general subgraph-based approach is taken where one may specify distributions of edges alongside distributions of arbitrary subgraphs. In the case of complete subgraphs it is obvious how to do this. For example, $G_{\boxtimes}$ subgraphs can be formed by allocating to nodes hyperstubs composed of three stubs. Then, four of these hyperstubs can be selected at random to form a $G_{\boxtimes}$ subgraph. However, it is not clear how this may work for subgraphs that are composed of more than one type of hyperstubs. For example, in a $G_{\boxslash}$, there are two different types of hyperstubs and it is necessary for any network model or construction algorithm to be able to make this distinction. Karrer and Newman proposed that it is possible to

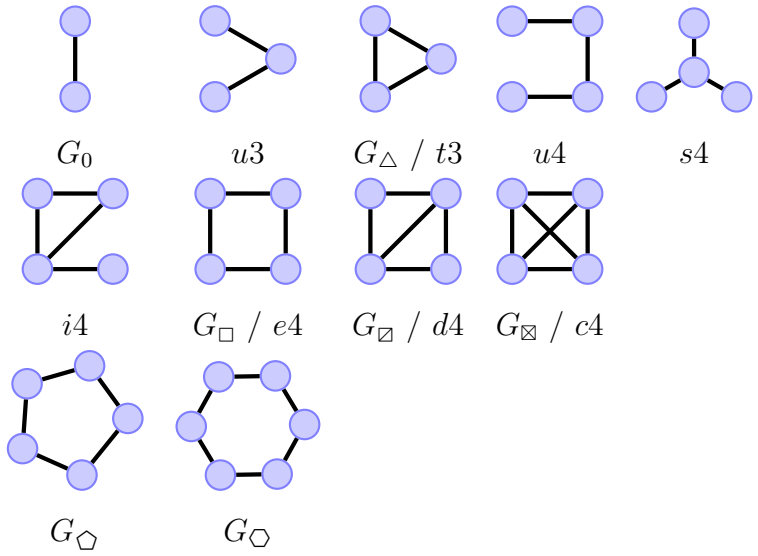


Figure 5.1: The set of subgraphs that have been used in this paper. The subgraphs denoted by: $\{G_0, G_\triangle, G_\square, G_\boxslash, G_\boxtimes, G_\pentagon, G_\hexagon\}$, are those that have been used as input for the proposed network construction algorithms. I use: $\{u3, t3, u4, s4, i4, e4, d4, c4\}$, to denote the total number of uniquely counted subgraphs given by the subgraph counting algorithm, see Chapter. 3.

identify a node's *role* within a subgraph using *orbits*. To find the orbits of a subgraph one must first list all possible automorphisms of the subgraph, that is, permutations of nodes that do not create or destroy edges. The orbit of a node is a set of other nodes with which it may be permuted so that no edges are created or destroyed. Of course, computing the automorphism group of subgraphs is computationally challenging but so long as subgraphs with few nodes are used, this is not a problem [32].

Network models are rarely used independently of other processes. Instead, they typically provide the substrate for dynamical processes to operate upon. For example, the compartmental Susceptible-Infected-Recovered (SIR) model of contagion is often embedded into a network to help better understand how the network and its properties affect the epidemic. Chapter 4 successfully incorporated the Karrer and Newman approach into an approximate ODE or mean-field model for SIR epidemics on networks displaying higher-order structure, and this mean-field model showed excellent agreement with simulation results. In order to achieve this, Ritchie et al. bypassed the need to classify a node's role in a subgraph via the automorphism group. Instead, nodes within arbitrary subgraphs were uniquely enumerated, even if they were topologically equivalent to one another, and this enumeration defined their role. The motivation for this adaptation was to simplify the derivation of the ODE model. Using the orbit approach or the full enumeration are different ways of satisfying different modelling needs, and these are not the only possible approaches. In fact, when modelling networks and nodes within subgraphs one can instead classify nodes by the stub cardinality of their hyperstubs.

A common method across all of the above models, i.e., edge-triangle, the more general Karrer-Newman model, and that proposed by Ritchie et al. (see [66]), is that sequences of hyperstubs must be specified for each and every subgraph that is to be included. From these sequences it is possible to recover the network's degree sequence by multiplying them by the stub cardinality of the hyperstub which they represent, and then adding the resulting sequences. Therefore the degree sequence of the network is a result of the construction of the network rather than a quantity that is controlled for. However, given that the degree sequence of the network is probably the single most important characteristic of a network, there is a need for methods that can generate

networks with a particular subgraph family and distribution yet preserve a given degree sequence. In [66], I recently showed that it is possible to constrain the hyperstub sequences so that the 1^{st} and 2^{nd} moments of the resulting degree sequence are controlled. In this paper, I go beyond this work and propose two generation algorithms that provide full control over the degree sequence and clustering.

The chapter is organised as follows. In Section 5.2, I describe in detail the two generation algorithms, including tuning of clustering. In Section 5.3, I validate the algorithms and I explore the diversity of the generated networks by comparing them to the widely used Big-V rewiring scheme, see Chapter. 3. I further analyse networks generated by using different subgraph families or distributions. Epidemic and complex contagion models are simulated on these networks and I show that degree distribution and global clustering alone are not sufficient to predict the outcome of these processes. Finally, I discuss extensions and further research questions relating to my work.

5.2 Materials and methods

In this section I propose two new algorithms, both of which are parametrised by a degree sequence and a set of subgraphs. The algorithms construct hyperstub degree sequences (from which the input degree sequence may be recovered exactly) that can be used in a modified configuration model style connection procedure to realise a network.

There are some caveats regarding the preservation of the input degree sequence that are common to all configuration-like models. Firstly it is necessary for a degree sequence to add to an even number to be graphical. If it does not, a stub must be created or destroyed to satisfy this constraint. In general, hyperstub degree sequences must add with multiplicity equal to the number of times they appear in their parent subgraphs, i.e., a triangle hyperstub sequence must be divisible by 3. When selecting stubs or hyperstubs at random to form subgraphs it is possible that self or multi-edges may form. The number of these events happening depends only on the average degree $\langle k \rangle$ and thus remains constant with network size. It is possible to simply delete self-edges or collapse multi-edges down to a single edge. If this approach is taken then the guiding degree sequence will be violated. Instead I disallow such connections by reselecting

nodes in the connection procedure until no self or multi-edges will be created by forming the subgraph. This is known as the *matching algorithm* [50]. Finally, it is possible for the process to be left with no option other than to add subgraphs over existing links or selecting multiple instances of the same node. In this case I completely reset the algorithm, regenerating hyperstub sequences and forming subsequent connections until a network is formed.

5.2.1 The underdetermined sampling algorithm – UDA

The concept underpinning this algorithm is that for each node there are combinations of hyperstubs that will satisfy its degree. For example, a node with $k = 3$ classical edges could form 3 single G_0 edges or 1 G_0 edge and 1 $G_{\triangle}$ hyperstub. The number of possible arrangements will depend on the degree of the node and number of input subgraphs. From these arrangements a single one is selected at random. For a given degree k this problem is equivalent to solving an underdetermined linear Diophantine equation equal to k subject to positivity constraints. The coefficients are given by the edge counts of the hyperstubs, that are induced by the input subgraphs, and the solution will give the number of each hyperstub so that the degree of the node is matched exactly.

To generate a network using this algorithm, let us assume that a degree sequence, $D = \{d_1, d_2, \ldots, d_N\} \in \mathbb{N}_0^{1\times N}$, and the set of subgraphs to be included in the network's construction, $G = \{G_1, G_2, \ldots, G_l\}$, is given. Then, for each subgraph I classify its hyperstubs by their edge cardinality. It is now possible to form a vector that has elements specifying the number of edges in each hyperstub. From this vector I take the unique elements. For example, the $G_{\not\square}$ subgraph will have a corresponding hyperstub vector of $\alpha = (2, 3)$. For a given degree k I must consider all possible hyperstubs and hyperstub combinations that yield a classical degree equal to k. To systematically list all such combinations, I first concatenate all the hyperstub vectors into a single vector, $\boldsymbol{\alpha}$, to be used as coefficients for the following linear underdetermined Diophantine equation

$$k = \alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_r x_r, \tag{5.1}$$

where $k = k_{min}, k_{min} + 1, \ldots, k_{max}$ and r denotes the number of eligible hyperstubs – a node with degree $k = 3$ can only go on to form subgraphs where the hyperstubs contain

no more than three edges –, for the given degree k and which is solved subject to the constraint $\boldsymbol{x} \in \mathbb{N}_0^r$. A solution $\boldsymbol{x}$ of this equation corresponds to the number of each type of hyperstubs required to result in a node of degree k. For example, if α_1 and α_2 take values 1 and 2 corresponding to hyperstubs of G_0 and $G_\triangle$ respectively and the degree of the node is $k = 5$, the Diophantine equation would take the form $5 = x_1 + 2x_2$ and the solution space of this equation is given by the pairs $(x_1, x_2) = \{(5, 0),\ (3, 1),\ (1, 2)\}$. In general these equations may be solved recursively by fixing a trial value $x_i = j$ and reducing the dimensionality of the equation by absorbing this term. This is repeated until the equation becomes of the standard form: $k' = \alpha_1 x_1 + \alpha_2 x_2$, which can be solved explicitly. A solution obtained this way will form a single solution of the original equation. This process is then repeated for a different starting trial solution, and since I seek only positive solutions and k is finite, the corresponding solution space has a finite number of elements. Matlab code for this process is available at `https://github.com/martinritchie/Network-generation-algorithms`.

Once the entire solution space for each degree has been found it is possible to start forming the hyperstub degree sequences. To proceed, the algorithm works sequentially through the degree sequence $D = \{d_1, d_2, \ldots, d_N\}$ of the N nodes, where $d_i \in \{k_{min}, k_{min} + 1, \ldots, k_{max}\}$. By selecting at random a solution from the solution space that corresponds to $k = d_i$, that specifies the hyperstub configuration, and by concatenating all the selected solutions for all the nodes a hyperstub degree sequence of dimension $h \times N$, where h denotes the total number of hyperstubs induced by the input subgraphs, is formed.

For incomplete subgraphs it is not possible to select solutions of the Diophantine equations' solution spaces at random. The reason for this is two-fold: (1) not all incomplete subgraphs are composed of equal quantities of each of their constituent hyperstubs, and (2) hyperstubs with lower stub cardinality will appear more frequently than hyperstubs of higher stub cardinality because hyperstubs with fewer edges can be more readily accommodated into the degree of a node. Problem (1) may be addressed by representing every hyperstub induced by a subgraph in the vector of coefficients opposed to grouping hyperstubs by their stub cardinality. Problem (2) may be addressed by decomposing hyperstubs generated in excess into simple/classical edges. It should be

noted that both of these methods will bias the resulting sequences but that this bias is only present when incomplete subgraphs are specified as input for the UDA. One advantage of the method I use is that it is possible to calculate the number of hyperstubs that will be decomposed back into stubs using *integer partitions*, and this is detailed in the Appendix 5.5.1. In particular the following result holds,

$$p(k,\alpha) = \sum_{m=1}^{\lfloor \frac{k}{\alpha} \rfloor} m\big[p(k-\alpha m)) - p(k-\alpha(m+1))\big],$$

where $p(k,\alpha)$ is the number of times that α appears in the partitions of k, with $\alpha \leq k$.

This may be used to compute the number of times certain hyperstubs appear. Returning to the example of the homogeneous networks with $k = 5$, generated with G_0 and $G_{\boxslash}$ there will be 4 counts of the double hyperstub generated for every 2 counts of the triple corner in the partition space, since 5 can be partitioned as

$$\{\{1,1,1,1,1\},\{2,1,1,1\},\{3,1,1\},\{4,1\},\{2,2,1\},\{2,3\},\{5\}\},$$

and $p(5,2) = 4$ and $p(5,3) = 2$. It should be noted that $p(k,\alpha)$ will count how many times α appears in all possible partitions of k. However, since it is not possible for either a double or triple corner to appear with a degree 4 or 5 hyperstub this will not affect the result. This simple number theoretic consideration shows a viable way in which bias can be quantified or measured.

Pseudo-code for the UDA algorithm is given in Appendix 5.5.2, and the Matlab code is available from `https://github.com/martinritchie/Network-generation-algorithms`.

A priori clustering calculation

The global clustering coefficient is defined as the ratio between the total number of triangles and the total number of connected triples of nodes $\triangle + \vee$, since each triangle contains 3 triples of nodes: $C = \frac{\triangle}{\triangle + \vee}$. It should be noted that each unique triangle is counted 6 times and each unique triple is counted twice. The number of triples incident to a node of degree k is given by $\triangle + \vee = k(k-1)$ since a node will form a triple with every pair of its neighbours and each triple is counted twice. The expected number of

triples for a node of degree k is therefore obtained by adding $P(K = k) \times k(k-1)$ over all degrees, where $P(K = k)$ is the probability of finding a node of degree k. The expected number of triangles incident to a node of degree k, $\langle \triangle_k \rangle$, may be obtained from the Diophantine equations' solution space associated with that degree. To do this, one needs to add all occurrences of triangle corners, regardless of which subgraph they belong to, from that solution space and divide by the number of solutions in that particular solution space, since solutions are selected uniformly at random. Finally I are in a position to compute the expected global clustering coefficient as

$$C = \sum_{k=2}^{k_{max}} \frac{\langle \triangle_k \rangle}{P(K = k) \times k(k-1).} \tag{5.2}$$

For example, let us consider the homogeneous network with $k = 5$ and the input subgraphs G_0 and $G_{\boxslash}$. These subgraphs induce the vector of coefficients $\boldsymbol{\alpha} = (1, 2, 3)$ that, for $k = 5$, has the following solution space

$$\begin{aligned} G_0: &\quad 5 \quad 3 \quad 2 \quad 1 \quad 0, \\ g_2: &\quad 0 \quad 1 \quad 0 \quad 2 \quad 1, \\ g_3: &\quad 0 \quad 0 \quad 1 \quad 0 \quad 1, \end{aligned}$$

where the rows give the number of each hyperstub, the columns give an individual solution and g_2 and g_3 denote the double and triple hyperstub of $G_{\boxslash}$ respectively. From this I may calculate the expected number of triangles $\langle \triangle_5 \rangle$. In this example I can see that on average for every g_3 corner the UDA algorithm will generate two g_2 corners. Since the excess g_2 corners will be decomposed into edges, one observes that g_2 and g_3 will be generated in equal quantities. So the expected number of g_2 is given by the expected number of g_3, e.g., $2/5$ per node. Since g_2 denotes a triangle corner, the number of g_2 corners also gives the total number of triangles, that is uniquely counted and per node. So the expected number of triangle per node is $12/5$, each triangle being counted 6 times, and this network will have a theoretical global clustering of $C = 0.12$. Computationally, I verify this by generating such networks with $N = 5000$, and find that the number of open triples and triangles is exactly $|\vee| = 100000$ and $|\triangle| = 12120$, resulting in a global clustering of 0.1212, as expected.

5.2.2 Cardinality matching – CMA

The cardinality matching algorithm (CMA) requires as input a degree sequence, a set of subgraphs and corresponding *subgraph sequences*, i.e., multiple sequences specifying to which and how many subgraphs nodes belong to. Note that these sequences are not yet allocated to nodes. The algorithm proceeds to allocate hyperstubs of subgraphs to nodes that have a sufficient number of stubs to accommodate the hyperstub degree. The algorithm outputs hyperstub degree sequences, from which the input degree sequence may be recovered exactly. This then can be used to realise a network based on a modification of the configuration model.

To generate a CMA network one needs to first decide on a degree sequence D, a subgraph set $G = \{G_1, G_2, \ldots, G_l\}$ and a set of subgraph sequences $S = \{S_1, S_2, \ldots, S_l\}$, where $S_j(k)$, with $j = 1, 2, \ldots, l$ and $k = 1, 2, \ldots, N$, gives the number of times a node will be part of a G_j subgraph without specifying the precise hyperstubs that connect the node to a G_j subgraph. My goal is to map the subgraph sequences into hyperstub sequences which can then be allocated to nodes that can accommodate them. From the hyperstub sequence, it is possible to work out the lower bound on the degree of nodes that can accommodate a specific hyperstub sequence. To complete this mapping one needs to differentiate between complete and incomplete subgraphs.

For complete subgraphs the subgraph sequence is identical to its hyperstub sequence since there is only one way or hyperstub by which a node can connect to such a subgraph. Thus, multiplying the hyperstub degree by the number of edges in the hyperstub will give us the lower bound on the degree of nodes that can accommodate the hyperstub sequence. For incomplete subgraphs the subgraph sequence does not specify how the node connects to the subgraph. Hence, I need to determine how the various hyperstubs are allocated to nodes. To see how to do this let us consider an arbitrary subgraph G with subgraph sequence S. Given that the subgraph has m distinct hyperstubs, let $p = (p_1, p_2, \ldots, p_m)$ be the vector of probabilities of picking different hyperstubs. I note that the values of p reflects the proportion of each hyperstub found in the subgraph. For example, $G_{\boxslash}$ has two distinct hyperstubs that both appear with multiplicity two, in this case $p = (1/2, 1/2)$. This will ensure that their numbers are balanced and subgraphs

can be formed.

Next, using the multinomial distribution corresponding to subgraph G, $M^G(s_i^G, P)$ where s_i^G denotes the subgraph sequence of index i (this is not yet a node label), I pick hyperstub types to transform the subgraph sequence into hyperstub degree. For each s_i^G this will result in a vector of length m specifying the exact number of each hyperstub. It is possible to concatenate all the resulting choices from all multinomial distributions $M^G(s_i^G, p)$, where $i = 1, 2, \ldots, N$ to form the following matrix

$$\begin{array}{c} \\ h_1^G \\ h_2^G \\ \vdots \\ h_m^G \end{array} \begin{array}{c} \begin{array}{cccc} s_1^G & s_2^G & \ldots & s_N^G \end{array} \\ \begin{pmatrix} h_1^G(1) & h_1^G(2) & \ldots & h_1^G(N) \\ h_2^G(1) & h_2^G(2) & \ldots & h_2^G(N) \\ \vdots & \ddots & \ddots & \vdots \\ h_m^G(1) & h_m^G(2) & \ldots & h_m^G(N) \end{pmatrix} \end{array} = H^G,$$

where $h_i^G(j)$ denotes the number of h_i hyperstubs contributing to the subgraph degree s_j^G. I now need to compute the total number of edges specified by each column of the above matrix or by the hyperstub degree. This is given by $H^G(i) = \sum_{j=1}^{m} |h_j^G| h_j^G(i)$ that denotes the total number of edges required by the subgraph degree s_i^G, and where $|h_j^G|$ represents the number of edges needed to form hyperstub j in subgraph G and $i = (1, 2, \ldots, N)$. This process needs to be repeated for each subgraph to be included in the networks construction, i.e., for each subgraph G_i with subgraph sequence $S^{G_i} = (s_1^{G_i}, s_2^{G_i}, \ldots, s_N^{G_i})$ there is a corresponding H^{G_i} with elements that the algorithm will use as the lower bound on the degree of the nodes that can accept such a selection of hyperstubs.

The algorithm then proceeds by choosing the largest values, $H_{\max}$, from all H^{G_i} matrices, and this is used as the lower bound on the degree of nodes that can accept the hyperstub configuration associated with $H_{\max}$, i.e., have enough edges of the classical type. From this list of all nodes with degree equal to or larger than $H_{\max}$, a node is selected uniformly at random. The degree of the selected node is reduced accordingly, and the index of the node is now associated with the hyperstub degree to $H_{\max}$. This node is then removed from the pool of eligible nodes for that particular subgraph, as otherwise it may be selected twice for the same subgraph thus violating the subgraph

degree sequence. Similarly, the element $H_{\max}$ is also removed from the pool of subgraph degree sequences that have yet to be allocated to nodes. This needs to be repeated until all elements of each subgraph degree sequence are allocated to nodes. Any edges that are not allocated to a particular hyperstub or subgraph are left to form edges.

In some cases it may be necessary to impose some cardinality constraints on the subgraph sequences. Obviously, if the network is homogeneous with $k = 3$ I cannot include complete pentagon subgraphs or allocate two $G_{\triangle}$ subgraphs to each node. More generally, it may be necessary to constrain the moments of the subgraph sequences. Let $\langle k \rangle$ denote the mean degree of the given degree sequence and let G_i be a subgraph composed of a single hyperstub with cardinality α and having subgraph degree sequence with mean $\langle s \rangle$ then: $\langle \alpha s \rangle = \alpha \langle s \rangle \leq \langle k \rangle$ is a necessary condition for the two sequences to be graphical. In the case of more than one hyperstub, this is extended to $\sum_{i=1}^{m} \alpha_i \langle s_i \rangle \leq \langle k \rangle$, where m, α_i and s_i denote the number of hyperstubs, hyperstub cardinality and associated subgraph sequence respectively. For the networks generated in this paper, the degree sequence and subgraph sequences were measured from networks previously generated by the UDA such that prior knowledge about the sequences being graphical was available without the need to impose any such constraints.

Clustering calculations for this algorithm are trivial since the subgraph degree sequences are known. One simply sums a sequence and then multiplies this figure by the number of triangles induced by that subgraph, being careful not to double count across multiple sequences for the same subgraph. The number of triples of connected nodes can be calculated following the method given for the UDA given in Section 5.2.1. Pseudo-code for the CMA is given in Appendix 5.5.3, with the corresponding Matlab code available from `https://github.com/martinritchie/Network-generation-algorithms`.

5.2.3 Connection process

I describe this process for a single incomplete subgraph. The case of the complete subgraph is trivial and has already been described (see Section 5.1). This process was first presented by Karrer & Newman [32]. Consider a subgraph composed of three different hyperstub types, h_1, h_2 and h_3 that occur with a multiplicity of 1, 2 and 3

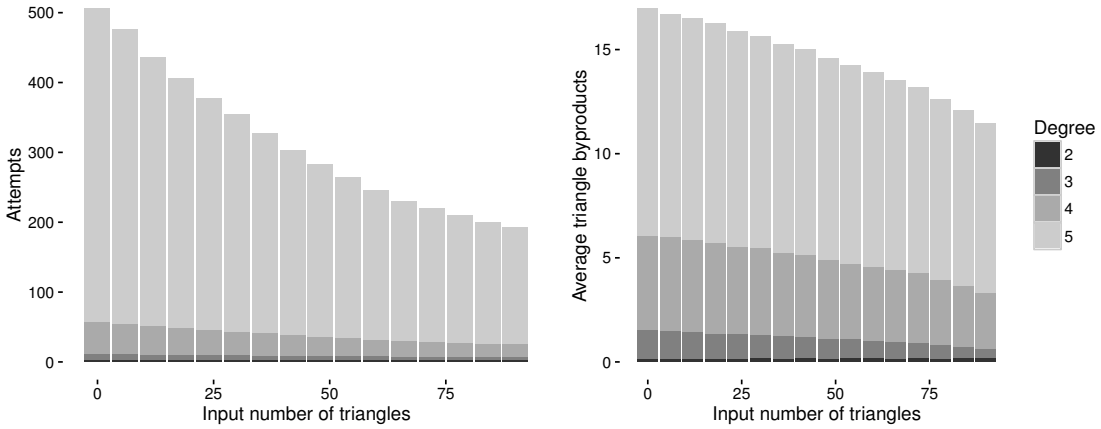
respectively, i.e., the subgraph is composed of 6 nodes. I require the following necessary conditions for the hyperstub sequences to be graphical

$$\sum_{i=1}^{N} |h_1|_i = \frac{1}{2}\sum_{i=1}^{N} |h_2|_i = \frac{1}{3}\sum_{i=1}^{N} |h_3|_i, \tag{5.3}$$

where $|h_i|_j$ specifies the h_i hyperstub degree of node j. If these conditions are not met, one needs to decompose any surplus hyperstubs into stubs that may form classical edges in order to preserve the degree sequence.

Using the hyperstub sequences, one can create three dynamic lists for the three hyperstub types where a node appears with multiplicity equal to its hyperstub degree. Once the dynamic lists are fully populated, the connections process can start. This is done by selecting the following: 1 node from the h_1 bin, 2 from the h_2 bin and 3 from the h_3 bin, and all the selection processes done uniformly at random and without replacement. Before forming the connections between these 6 nodes, one must ensure that: (1) the selection contains no duplicates (that will form self-edges) and (2) that no single pair of nodes are already connected. If a connection already exists, a multi-edge may form and/or subgraphs will share edges. If neither of these conditions are violated then the connections may be formed. Otherwise all nodes are returned to their bins and a new selection is made. It is possible that after many selections no valid combinations of nodes remain. For example, all bins may contain the same node. In this and other non-viable cases, all bins are re-populated and the connection process is started anew.

As previously discussed, it is possible to delete self and multi-edges but this will destroy the degree sequence. The method of reselecting nodes has been previously introduced and is known as the *matching algorithm* see [50]. However, it has previously been shown that the matching algorithm introduces a bias when constructing networks [39]. Ideally, when a self or multi-edge is formed one would start the whole connection process from scratch, the so-called *refusing algorithm*. This results in an unbiased sampling [39]. For the configuration model the number of such self and multi-edges depends on the first and second moments of the degree distribution [56]. As such, an unbiased configuration model approach may result in prohibitive running times as $\langle k \rangle$ increases.



(a) Attempts (b) Triangle by-products

Figure 5.2: (a) The average number of *refusal* attempts to realise a network and (b) the average number of triangle by-products found per networks. In both cases the CMA was parametrised with only G_0 and $G_\triangle$ subgraphs. As more $G_\triangle$ are specified as input both the number of average number of attempts and triangle by-products decreases. Triangle by-products are computed by subtracting the input from the measured number of triangles.

Currently, there are no analytical results regarding the probability of self or multi-edges as well as bias for the subgraph connection process. To help develop some understanding I set up the following experiment: using the CMA and the refusing algorithm I generate a series of homogeneous networks. Initially the CMA is parametrised with no $G_\triangle$ subgraphs and only G_0, reverting to the configuration model. For increasing degree of $k = 2, 3, 4, 5$ I then determine the average number of attempts required before a network is produced as well as the average number of $G_\triangle$ by-products. I then repeat this but with the CMA parametrised with an increasing number of $G_\triangle$ subgraphs, distributed so that a node is incident to at most one $G_\triangle$ subgraphs, and so on. Figure 5.2 illustrates that both the number of attempts and that of $G_\triangle$ by-products per network increase with degree, as one would expect. It also reveals that these quantities decrease when the CMA is parametrised with increasing numbers of $G_\triangle$ subgraphs, regardless of degree. Since the number of attempts per networks is a function of the number of self and multi-edges, these results also imply that the number of self and multi-edges reduce as the number of $G_\triangle$ increase.

I believe the following to be an intuition behind this surprising result: consider a node incident to two $G_\triangle$ stubs. For a self loop to be created about this node there is a single opportunity: both hyperstubs must be simultaneously selected during the connection procedure. Now, if I consider a node with the same degree, but with each of its of stubs being used to form only lines, then there are $k(k-1)/2 = 6$ different ways in which pairs of stubs may be selected that result in a self edge. Thus, in general, hyperstubs will reduce the number of ways in which tuples of nodes may be connected, compared to stubs, and this will impact both the self and multi-edge probabilities.

Edge probability: With the subgraph connection process it is possible to replicate some of the estimates for the number of self and multi-edges that exist for the standard configuration model, as shown in [56]. The following calculations are intended to further develop intuition and by no means form a rigorous argument. Let us consider a network model composed of only $G_\triangle$ subgraphs, referred to henceforth as the $G_\triangle$ model. Let $3m_t$ denote the total number of $G_\triangle$ hyperstubs, i.e, this network has a total of m_t $G_\triangle$ subgraphs and $6m_t = 2m$ stubs, since each $G_\triangle$ hyperstub is composed of two stubs.

I first consider the probability of two nodes sharing a single edge in the $G_\triangle$ model.

Let nodes i and j have $G_\triangle$ degrees of t_i and t_j respectively. A single hyperstub of i may connect to any of the t_j hyperstubs originating from j. The probability of selecting one of j's hyperstubs is $t_j/(3m_t - 1)$, since I can no longer select the initial hyperstub incident to i. However, any one of i's hyperstubs could connect to any one of j's hyperstubs, and $t_i t_j/(3m_t - 1)$ correctly accounts for this. A third hyperstub must now be selected, incident to a third distinct node, i.e., any one of the $3m_t - t_i - t_j$ hyperstubs that are not incident to either i or j. If a hyperstub incident to i or j were to be selected this would result in both a self and multi-edge. Therefore, the probability of i and j sharing a single edge is given by

$$p_{i,j} = \frac{t_i t_j}{3m_t - 1}\left(\frac{3m_t - t_i - t_j}{3m_t - 2}\right) \tag{5.4}$$

Since the degree distribution is fixed and I are interested in the limit as $N \to \infty \Rightarrow m_t \to \infty$,

$$\lim_{N\to\infty} p_{i,j} = \frac{t_i t_j}{3m_t}. \tag{5.5}$$

Let us now consider that in the $G_\triangle$ model each node is incident to $2t_i = k_i$ stubs in a network composed of a total of $2(3m_t) = 2m$ stubs. By making these substitutions into equation 5.5 the edge probability of the $G_\triangle$ model can be compared to its equivalent configuration model

$$\frac{\frac{k_i}{2}\frac{k_j}{2}}{6m_t} = \frac{k_i k_j}{4m} < \frac{k_i k_j}{2m},$$

where the r.h.s. represents the edge probability in the configuration model. This counter-intuitive result for the $G_\triangle$ model is due to half of a node's stubs being obliged to connect to a third distinct node, excluding possibilities of i and j connecting which would otherwise be possible in the configuration model.

Multi-edge expectation: As in the standard configuration model I can use equation (5.5) to estimate the number of multi edges which may happen in two ways when selecting a triplet of nodes: (a) atleast one of the constituent pairs already being connected or (b) all three of nodes already being connected. I first consider (a), the more likely scenario. i and j will share an edge with the probability given in equation (5.5).

To compute the probability of finding a second edge between nodes i and j in the $G_\triangle$ model one must compound $(t_i-1)(t_j-1)/(3m_t-1)$ with equation (5.5)

$$P(A(i,j)>1) \;=\; \frac{t_i t_j (t_i-1)(t_j-1)}{(3m_t)^2},$$

adding this probability over all pairings of nodes and dividing by 2 to remove the double count, yielding

$$\begin{aligned}\lim_{N\to\infty}\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}p(A(i,j)>1) &= \lim_{N\to\infty}\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{t_i t_j (t_i-1)(t_j-1)}{(3m_t)^2} \\ &= \lim_{N\to\infty}\frac{1}{2}\sum_{j=1}^{N}\frac{t_j(t_j-1)}{3m_t}\sum_{i=1}^{N}\frac{t_i(t_i-1)}{3m_t} \\ &= \frac{1}{2}\left(\frac{\langle G_\triangle^2\rangle-\langle G_\triangle\rangle}{\langle G_\triangle\rangle}\right)^2,\end{aligned} \tag{5.6}$$

where I have used:

$$3m_t=\langle G_\triangle\rangle N,\quad \langle G_\triangle\rangle=\frac{1}{N}\sum_{i=1}^{N}t_i,\quad \langle G_\triangle^2\rangle=\frac{1}{N}\sum_{i=1}^{N}t_i^2. \tag{5.7}$$

Again, compare this value to that of the standard configuration model with the substitutions $2t_i=k_i$ and $2(3m_t)=2m$ yielding

$$\frac{1}{2}\left(\frac{\frac{1}{2}\langle k^2\rangle-\langle k\rangle}{\langle k\rangle}\right)^2<\frac{1}{2}\left(\frac{\langle k^2\rangle-\langle k\rangle}{\langle k\rangle}\right)^2,$$

where the r.h.s. represents Newman's original estimate for multi-edges in the configuration model [56]. Now consider scenario (b), selecting the same triplet of nodes twice resulting in 3 multi edges. Consider the nodes $i,\ j$ and l with $G_\triangle$ degrees of $t_i,\ t_j$ and t_l. In the limit of large networks this triple of nodes are connected with probability

$$\lim_{N\to\infty}p_{i,j,l}=\lim_{N\to\infty}\frac{t_i t_j t_l}{3m_t(3m_t-1)}=\lim_{N\to\infty}\frac{t_i t_j t_l}{9m_t^2},$$

the probability of this triple being selected twice is approximately

$$\lim_{N\to\infty}\frac{t_i t_j t_l (t_i-1)(t_j-1)(t_l-1)}{(3m_t)^4}.$$

This probability can be added over all triplets of nodes yielding

$$\begin{aligned}
&\lim_{N\to\infty}\frac{1}{3}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{N}\frac{t_i t_j t_l(t_i-1)(t_j-1)(t_l-1)}{(3m_t)^4}\\
=\;&\lim_{N\to\infty}\frac{1}{3(3m_t)}\sum_{i=1}^{N}\frac{t_i(t_i-1)}{3m_t}\sum_{j=1}^{N}\frac{t_j(t_j-1)}{3m_t}\sum_{l=1}^{N}\frac{t_l(t_l-1)}{3m_t}\\
=\;&\lim_{N\to\infty}\frac{1}{3N\langle G_\triangle\rangle}\left(\frac{\langle G_\triangle^2\rangle-\langle G_\triangle\rangle}{\langle G_\triangle\rangle}\right)^3, \qquad (5.8)
\end{aligned}$$

where I have again used equations (5.7). The expected number of multi-edges created by two $G_\triangle$ subgraphs connected on the same triplet of nodes is not constant with network size but instead tends to zero with increasing network size. This result, alongside equation (5.6), suggests that the number multi-edges in the $G_\triangle$ model will be less than what is found in the equivalent configuration model network. I next consider the probability of a self edge in the $G_\triangle$ model.

The number of self-edges: During the connection process of the configuration model self edges are created when two stubs that are incident to the same node are connected. The analogue of this in the $G_\triangle$ model is selecting three hyperstubs incident to the same node, resulting in three self-edges. I shall denote this event $\{i,i,i\}$

$$\begin{aligned}
p(\{i,i,i\}\wedge\{i,i,i\}) &= \frac{\binom{t_i}{3}}{(3m_t)(3m_t-1)},\\
\lim_{N\to\infty}p(\{i,i,i\}\wedge\{i,i,i\}) &= \lim_{N\to\infty}\frac{t_i(t_i-1)(t_i-2)}{6(3m_t)^2},
\end{aligned}$$

this value can be added over all nodes to estimate the expected number of self-edges in the network

$$\lim_{N\to\infty}\sum_{i=1}^{N}\frac{t_i(t_i-1)(t_i-2)}{6(3m_t)^2} = \frac{\langle G_\triangle^3\rangle-3\langle G_\triangle^2\rangle+2\langle G_\triangle\rangle}{6N\langle G_\triangle\rangle^2}, \qquad (5.9)$$

where I have used equations (5.7). This value, like equation (5.8) is not fixed with network size and instead tends to zero as N becomes large.

Node duplicates: In the $G_\triangle$ model it is possible to select a pair of hyperstubs incident to the same node alongside a distinct third node, resulting in a self and multi-edge. I

shall denote this event $\{i, i, j\}$. Then

$$\begin{aligned} \lim_{N\to\infty} p(\{i,i,j\}) &= \lim_{N\to\infty} \left(\frac{\binom{t_i}{2}}{3m_t} \left(\frac{3m_t - t_i}{3m_t - 2} \right) \right) \\ &= \lim_{N\to\infty} \frac{t_i(t_i-1)}{2(3m_t)}, \end{aligned} \tag{5.10}$$

which, after adding over all nodes, yields

$$\lim_{N\to\infty} \sum_{i=1}^{N} \frac{t_i(t_i-1)}{3m_t} = \frac{\langle G_{\triangle}^2 \rangle - \langle G_{\triangle} \rangle}{2\langle G_{\triangle} \rangle} \tag{5.11}$$

Since the determining factor of this expectation is the selection of a pair of hyper-stubs incident to the same node I shall compare it to the self-edge probability for the equivalent configuration model network

$$\begin{aligned} \frac{\langle G_{\triangle}^2 \rangle - \langle G_{\triangle} \rangle}{2\langle G_{\triangle} \rangle} &= \frac{\langle (k/2)^2 \rangle - \langle k/2 \rangle}{2\langle k/2 \rangle} \\ &= \frac{\frac{1}{4}\langle k^2 \rangle - \langle k \rangle}{\langle k \rangle} \\ &< \frac{\langle k^2 \rangle - \langle k \rangle}{2\langle k \rangle}, \end{aligned} \tag{5.12}$$

i.e., as $N \to \infty$ I expect that the number of duplicate node selections resulting from $G_{\triangle}$ placement will be strictly less than the number of self-edges in the equivalent configuration model network.

By-products: It is possible for previously created subgraphs to become connected into a set of subgraphs with overlap, see Figure 5.3 for an illustration. The expected number of multi-edges above demonstrates the possibility for one such occurrence, here, two $G_{\triangle}$ subgraphs sharing an edge. In this case if the multi-edge was collapsed down to a single edge the process would yield a $G_{\boxslash}$ subgraph. The expected number of these events was shown to be bounded by the number of multi-edges in the equivalent configuration model network. However, this type of connection was not permitted in my implementation.

Currently, I are unable to offer estimates regarding the frequency of erroneous $G_{\triangle}$ subgraphs, that is, $G_{\triangle}$ subgraphs that appear beyond that which were controlled for.

This type of connection is permitted in my implementation and would result in the subgraph by-products as shown in Figure 5.3. However, Figure 5.2 indicates that the number of erroneous $G_{\triangle}$ subgraphs decreases as the number of intended subgraphs increases.

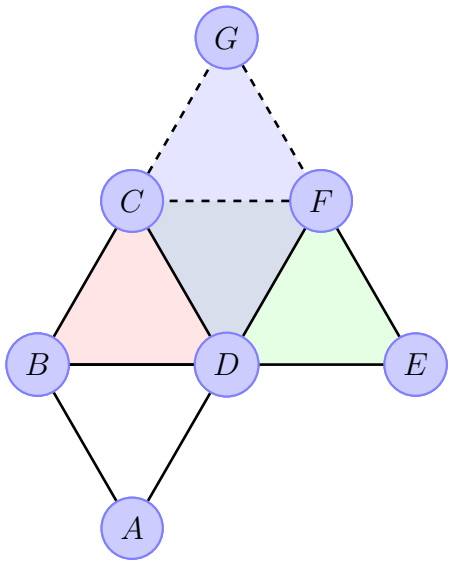


Figure 5.3: Unintended generation of subgraphs with overlap. Despite satisfying the generation constraints given in Section 5.2.3, the addition of triangle (C,G,F) to toast (A,B,C,D) and triangle (D,F,E) results in 3 unintended distinct toasts {(B,C,F,D) in red, (D,C,F,E) in green, and (D,C,G,F) in blue} overlapping on one unintended triangle (C,F,D), in gray.

5.2.4 Models of contagion

In order to illustrate the impact of network structure – and higher-order structure particularly – different epidemic dynamics were simulated on the generated networks. Three different models were chosen: Susceptible-Infected-Susceptible (SIS), Susceptible-Infected-Recovered (SIR) and complex contagion [47, 60]. To simulate SIS and SIR dynamics, the fully susceptible network of nodes is perturbed by infecting a small number of nodes. Infected nodes spread the infection to susceptible

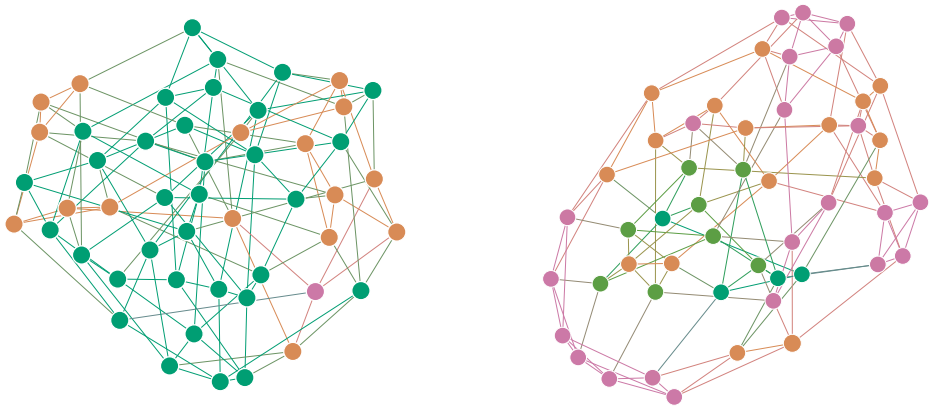
neighbours at a per-link rate of infection τ. Infected individuals recover independently of the network at rate γ and become susceptible again (for SIS dynamics) or become removed (for SIR epidemics). In contrast to the infection process in the previous two dynamics, the complex contagion process requires that susceptible nodes are exposed to multiple infectious events before becoming infected. These events must be from different infectious neighbours as only the first infection attempt from an infectious node counts. This critical infection threshold for each node is set in advance and is usually bounded from above by the degree of the node. To simulate the complex contagion dynamics, nodes are allocated infection thresholds $r_i \in \mathbb{N}$, where $i = 1, 2 \ldots, N$, and the fully susceptible population of nodes is perturbed by infecting an initial number of nodes chosen at random. In this model a susceptible node i becomes infected as soon as it has received atleast r_i infectious contacts from r_i distinct infectious neighbours. There is no recovery in this model and infected individuals remain infected for the duration of the epidemic.

5.3 Results

5.3.1 Algorithm validation

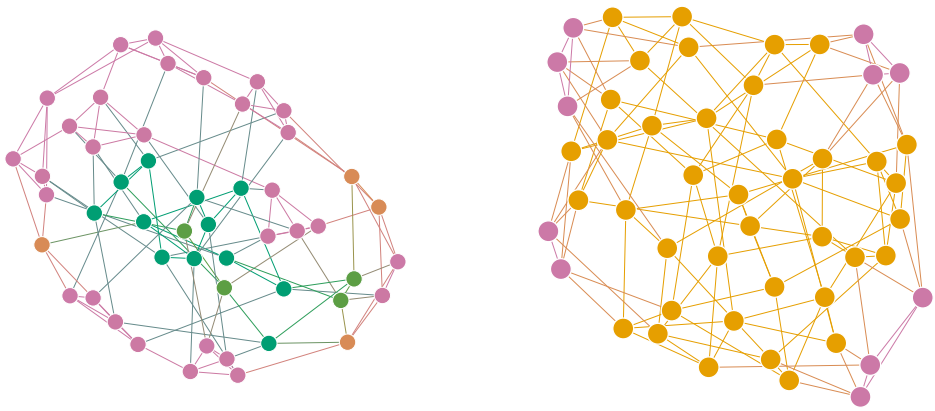
To validate my algorithms, I generated a number of networks with pre-specified degree distribution and subgraph set, as well as a multinomial distribution of subgraph corners or hyperstubs around nodes. I verified that the networks generated were as expected given the input.

As described in Section 5.2 the algorithms preserve the degree sequence, permitting at most a single edge to be deleted if the degree sequence sums to an odd number. The ability to exercise control over the networks' subgraph topology is illustrated by Figure 5.4. Note that Figure 5.4a shows a *random* network that includes $G_{\triangle}$ subgraphs. When constructing networks using the configuration model it is possible to create $G_{\triangle}$ subgraphs with non-zero probability and this is to be expected [57]. However, this is a function of mean degree not network size, and this probability goes to zero with network size going to infinity.



(a) Random

(b) Big-V, C=0.22



(c) UDA, C=0.22

(d) CMA, C=0.22

Figure 5.4: Small networks generated by the Big-V, UDA and CMA algorithms. All networks have the same homogeneous degree sequence with $k = 5$. The Big-V algorithm re-wired the random network, Figure 5.4a. The UDA was parametrised with subgraphs G_0 $G_{\boxslash}$ and $G_{\boxtimes}$. The CMA was parametrised so that every node was incident to 2 $G_{\triangle}$. The Big-V, UDA, and CMA networks all have a global clustering coefficient of $C = 0.22$. The network nodes are coloured so that green/orange/pink denotes nodes of low/medium/high clustering, respectively.

	c4	d4	e4	i4	s4	t3	u3	u4
Random	0	0	42	17	446	6	482	1706
Big-V	1	23	10	10	212	7	386	1220
UDA	7	10	22	5	243	1	389	1239
CMA	0	9	10	40	185	24	389	1201

Table 5.1: Subgraph counts for the networks of Figure 5.4. Note: if one adds a single $G_{\triangle}$ so that it shares a single edge with a $G_{\boxslash}$ and this edge is not the diagonal edge of $G_{\boxslash}$, then $d4$ increases by one but $t3$ will have only increased by one, not two. I note that $2 \cdot d3$ yields the maximum number of possible $G_{\triangle}$ induced by $G_{\boxslash}$. In general, calculating the number of $G_{\triangle}$ in this way will always yield the maximum possible count but not necessarily the true count because a single $G_{\triangle}$ could be shared by more than one $G_{\boxslash}$.

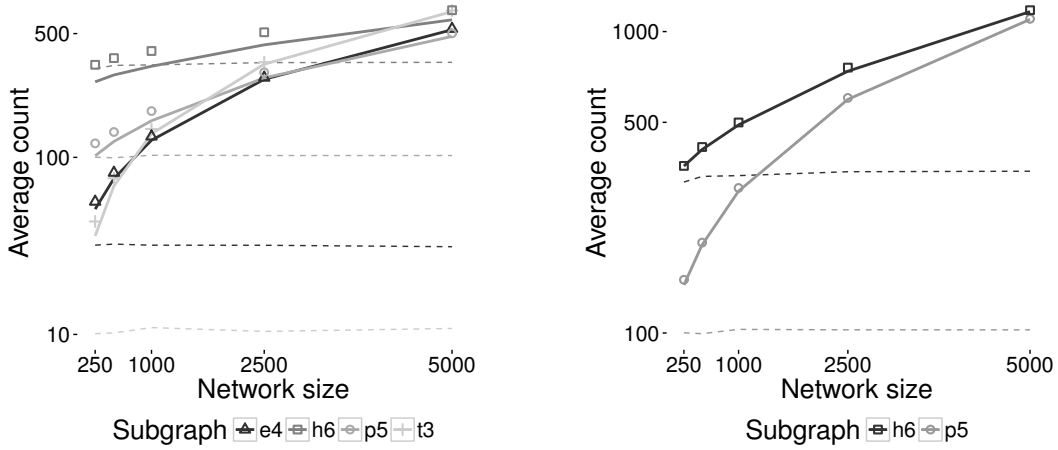
To properly demonstrate the proposed algorithms' control over the building blocks in the network, I shall use the subgraph counting algorithm, presented in Chapter. 3, to count the number of subgraphs *a posteriori*. In this implementation I counted subgraphs composed of 4 nodes or less – see the top two rows of Figure 5.1, as well as 5- and 6-cycles. Table 5.1 provides the subgraph counts for the networks displayed in Figure 5.4. It confirms that the random network given in Figure 5.4a contains 6 $G_{\triangle}$, counted uniquely, as observed above. The table also reveals that, through increasing the frequency of $G_{\triangle}$, the Big-V algorithm also introduced $G_{\boxslash}$ and $G_{\boxtimes}$ subgraphs. The UDA was parametrised with $\{G_0, G_{\boxslash}, G_{\boxtimes}\}$ and the table confirms a significant presence of these subgraphs when compared to the random network. Although the CMA was parametrised solely with $G_{\triangle}$ subgraphs distributed so that each node was incident to 2 $G_{\triangle}$ subgraphs, the subgraph counts reveal that this network contains 9 $G_{\boxslash}$ subgraphs. This is a consequence of attempting to generate *small* networks with such a high prevalence of triangles: it is highly likely that the algorithms will select nodes that already share one other common neighbour later in the connection process. One expects the proportion of these events to become increasingly negligible with greater network size.

Next, I used the above motif counting algorithm to evaluate the extent to which

the proposed algorithms can exert control over the prevalence of subgraphs in the generated networks. Figure 5.5 compares *measured* counts of subgraphs in UDA and CMA networks with *expected* counts. Here, an important observation must be made at the outset. Even in random networks, cycles ($G_{\square}$, $G_{\pentagon}$ and $G_{\hexagon}$) appear in significant quantities: 33, 100 and 333 times respectively, and regardless of network size. They are a natural consequence of the fact that the probability of selecting two nodes in different branches of a finite tree-like network is non-zero. Therefore, my *expected* counts are the sum of the counts expected *by construction* and those *measured* in the random networks. For example, since the CMA networks were generated with each node being incident to a single $G_{\hexagon}$ subgraph, a total of 833 uniquely counted $G_{\hexagon}$ subgraphs were expected *by construction* in networks of size $N = 5000$. However, because an average of 344 $G_{\hexagon}$ subgraphs were counted in random networks of size $N = 5000$, my *expected* count was $833 + 344 = 1177$. The *measured* count was found to be 1165. More generally, I found the *expected* counts to match well with the *measured* counts, indicating that the generating algorithms did not create by-products in addition to those observed at random[1]. However, these results also suggest that the level of control exerted by the algorithms over subgraph prevalence depends on how often those subgraphs appear naturally as by-products. Control is strongest for subgraphs that do not appear naturally as by-products. When considering subgraphs that appear naturally with high frequency, e.g., $G_{\pentagon}$, real control over their prevalence can only be achieved if an even higher frequency is imposed, which may not always be possible for a given degree sequence and global clustering.

In what follows, I set out to highlight differences between the new algorithms compared to classic ones and also to emphasise the diversity within networks generated by the same algorithms.

[1] Although I will show in Section 5.3.3 that for specific parameterisations of CMA, by-products are possible.



(a) UDA

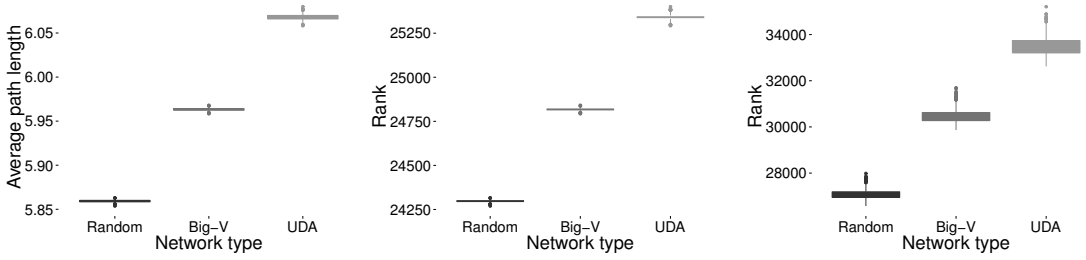
(b) CMA

Figure 5.5: A comparison of subgraphs found in the UDA and CMA networks to their random network analogues and expected counts plotted with thick lines, thin lines and discrete markers respectively. $p5$ and $h6$ denote the counts of $G_{\pentagon}$ and $G_{\hexagon}$ respectively. All networks have the same homogeneous degree sequence with $k = 5$ but with increasing size: $N = 250, 500, 1000, 2500, 5000$, where 100 of each size was generated. (a) The UDA algorithm was parametrised with subgraphs $\{G_{\triangle}, G_{\square}, G_{\pentagon}, G_{\hexagon}\}$, and the resulting average subgraph counts are shown on the left. (b) The CMA algorithm was parametrised so that each node was incident to a single $G_{\pentagon}$ and $G_{\hexagon}$ subgraph, and the resulting average subgraph counts are shown on the right. The expected values were calculated by adding the total counts from the subgraph sequences, dividing them by the subgraphs' node cardinality, and adding these figures to the number of subgraphs found as by-products in the random networks.

5.3.2 Sampling from a different area of the network state space

In this section, I seek to highlight the versatility of the proposed generation mechanisms by showing that, given a degree distribution and a global clustering, they sample different areas of the network state space than existing methods such as Big-V. I begin by reminding the reader that the Big-V algorithm searches for paths of 5 nodes and rewires such paths so that additional triangles are created. In other words, the principal building block of this algorithm is the $G_{\triangle}$ subgraph and subgraphs that may be constructed by overlapping $G_{\triangle}$ subgraphs. It follows that this algorithm is unlikely to give rise to a higher than expected at random number of $G_{\square}$ or other 'empty' cycles. The UDA algorithm was therefore parametrised with subgraph family $\{G_0, G_{\triangle}, G_{\square}, G_{\pentagon}, G_{\hexagon}\}$. In order to eliminate the effect of degree heterogeneity, a homogeneous degree sequence with $k = 5$ was used. The resulting networks had a global clustering coefficient of $C = 0.04$, induced by 666 (uniquely counted) $G_{\triangle}$ subgraphs. I then used the Big-V algorithm to rewire random networks constructed using the same degree sequence until the desired level of clustering, $C = 0.04$, was achieved. Significant differences between generated networks would confirm that the Big-V and UDA generated networks are sampled from different areas of the state space of networks satisfying that degree sequence and global clustering. As a further point of reference, data taken from a random network realisation, generated using the configuration model, of the degree sequence was included in all of my analyses. Henceforth I shall refer to these three types of networks as network family **A**.

In Figure 5.6, the distributions of the average path length, average betweenness centrality and maximum betweenness centrality for the above networks are given. In general, an increase in clustering results in a higher value of the average path length – see the average path length of random and Big-V networks in Figure 5.6a. This is a known result [6]. Surprisingly, a similar magnitude of difference in average path length and average and maximum betweenness centrality is observed between the Big-V and UDA networks despite them having the same global clustering, see Figure 5.6a, 5.6b and 5.6c, respectively. Output from the subgraph counting algorithm (Figure 5.7) confirms that, as expected, the Big-V algorithm does not generate more $G_{\square}$ subgraphs



(a) Average path length (b) Average betweenness (c) Maximum betweenness

Figure 5.6: Plots of the average path length and diameter for homogeneous networks ($N = 5000$ and $k = 5$) for network family **A**. The Big-V algorithm was parametrised solely by clustering, in this case $C = 0.04$, to best suit the networks produced by the UDA. The differences in average path length, average betweenness centrality and maximum betweenness centrality between the random network and its Big-V analogue were of similar magnitude as the differences between the Big-V network and the cycle-based UDA networks, and these were significant.

than are observed in the random network. More generally, the results show that the Big-V and UDA networks exhibit markedly different subgraph topologies with the Big-V networks relying heavily on $G_{\boxslash}$ to cluster the networks unlike UDA networks that rely almost exclusively on $G_{\triangle}$ not appearing as part of any other subgraph. It may be that such variation was facilitated by the low level of clustering considered, and that with higher clustering, eliciting such differences might be more challenging. However, these results provide evidence that the UDA algorithm can sample from a different part of the state space than the Big-V algorithm.

5.3.3 Diversity within the newly proposed algorithms

In this section, I illustrate the diversity of networks generated with UDA and CMA by exploring the impact of subgraph distribution over nodes (for identical degree distribution and global clustering) and how it may change network characteristics.

To do this I first parametrised the UDA with subgraph family $\{G_0, G_{\triangle}, G_{\square}, G_{\boxslash}, G_{\boxtimes}\}$ (chosen due to its frequent use in the literature, e.g., [6, 23, 28, 32, 65, 66]), and a

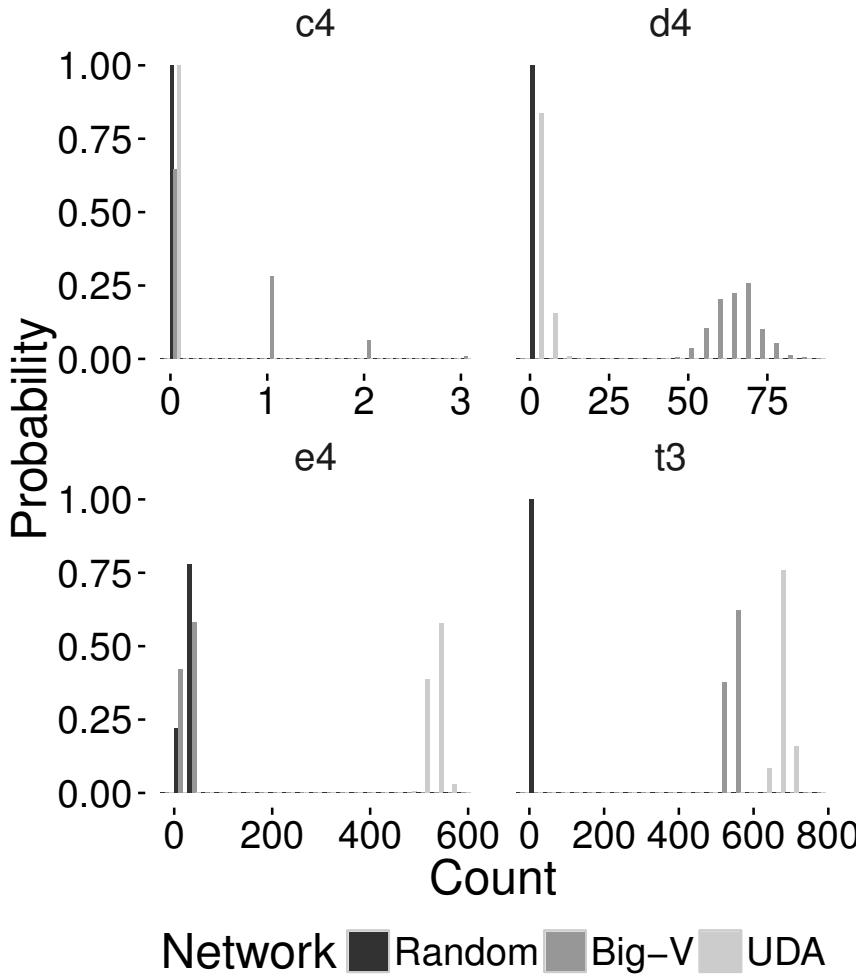
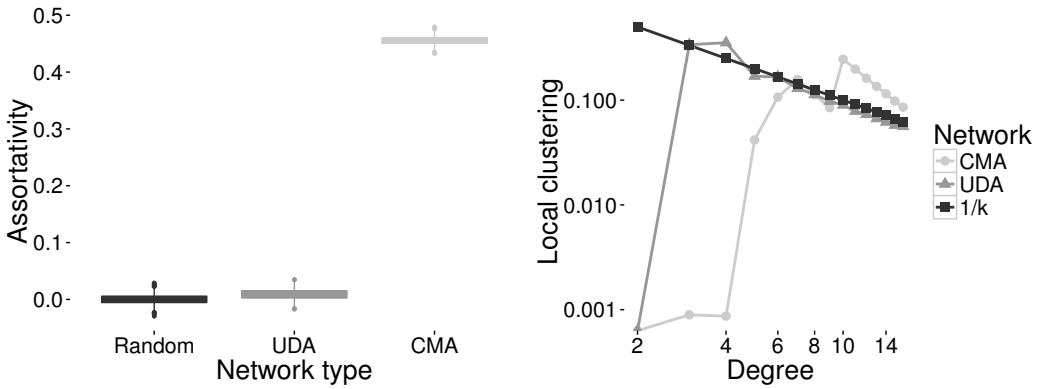


Figure 5.7: Distributions of total number of subgraphs in network family **A** ($N = 5000$, $k = 5$). The Big-V and UDA networks have a global clustering coefficient of $C = 0.04$. All given counts are unique. The $t3$ counts denote the number of $G_{\triangle}$ subgraphs that are not involved in any subgraphs of four nodes (i.e., $G_{\boxslash}$ and $G_{\boxtimes}$). However, the $c4$ and $d4$ counts may include $G_{\triangle}$ subgraphs shared by $G_{\boxslash}$ and $G_{\boxtimes}$. The number of $G_{\square}$ subgraphs generated by the Big-V algorithm is very close to the counts found in random networks.

heterogeneous degree sequence generated using the Poisson distribution with $\lambda = 5$. Since it is difficult to control for the number of subgraphs that appear in a network generated using the UDA I counted the total number of each subgraph, from UDA-produced subgraph sequences, and used these counts to create alternative subgraph sequences as input to the CMA, see Section 5.2.2, rather than drawing such sequences from a theoretical distribution. The resulting networks were therefore expected to have identical degree sequence, global clustering of 0.13 and subgraph counts. Since the CMA allows us to choose arbitrary sequences of subgraphs, I opted to push the clustered subgraphs, $\{G_{\triangle}, G_{\boxslash}, G_{\boxtimes}\}$, onto the higher-degree nodes to accentuate the effect of clustering. I did this by specifying that these subgraphs had to appear with multiplicity greater than one. For example, a degree-three $G_{\boxtimes}$ hyperstub required a minimum $k = 9$-degree node. As previously, I included a random network realisation of the heterogeneous degree sequence for comparison. Henceforth, I shall refer to these three types of networks as network family **B**.

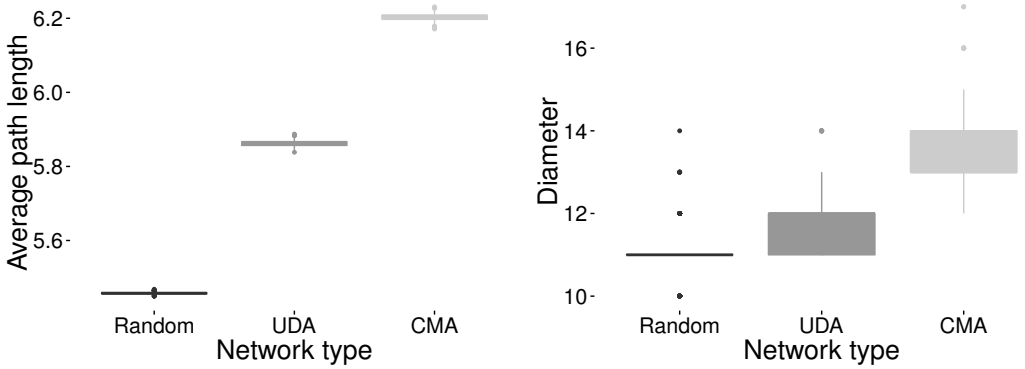
The heterogeneity in degree distribution allows us to use additional degree-dependent metrics: degree-degree correlations and degree-dependent clustering [53, 68]. These have been plotted in Figure 5.8. The plot for the degree-degree correlation coefficient shows that by aggregating clustered subgraphs around high-degree nodes, the CMA-constructed networks yield a higher assortativity than that of UDA and random networks, see Figure 5.8a. This is an important property of the methodology since the clustering potential of a network is bounded by the degree-degree correlation coefficient [68]. Moreover, if one wishes to maximise clustering in heterogeneous networks, it is necessary for nodes of similar degree to mix preferentially. Figure 5.8b shows that the CMA networks yield a negatively skewed distribution of degree-dependent clustering, with nodes of degree $k \geq 9$ contributing most to clustering. The ability to manipulate the degree and clustering relationship as well as assortativity clearly demonstrates the broader scope of the CMA when sampling from the ensemble of networks with same degree distribution and global clustering.

As with network family **A**, an increase in average path length, diameter, average and maximum betweenness centrality of UDA and CMA networks over random networks will be attributable to the increased global clustering coefficient, $C = 0.13$, see Figures 5.9



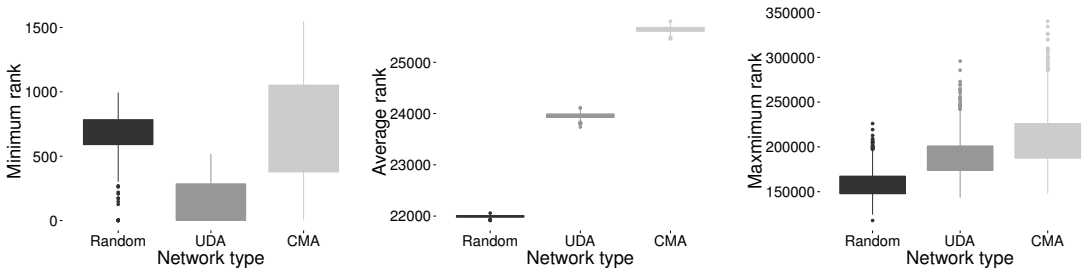
(a) Assortativity (b) Degree-dependent clustering

Figure 5.8: Plots of assortativity and degree-dependent average local clustering for network family **B** with $k \sim Pois(5)$. The UDA and CMA networks have a global clustering coefficient of $C = 0.13$. The distribution of subgraphs in CMA networks was manipulated so that the clustered subgraphs $\{G_{\triangle}, G_{\boxslash}, G_{\boxtimes}\}$ appeared around nodes with multiplicity greater than one. In order to preserve the subgraph degree sequence these aggregated subgraphs were allocated to the higher degree nodes, resulting in higher assortativity and a more positively skewed distribution of degree-dependent clustering.



(a) Average path length (b) Diameter

Figure 5.9: Plots of average path length and diameter for network family **B** with $k \sim Pois(5)$. The UDA and CMA networks have a global clustering coefficient of $C = 0.13$. The increased average path length and diameter between the UDA and random networks is attributable to the higher clustering. The similar increase between UDA and CMA networks is a reflection of the higher assortativity of the CMA networks.



(a) Minimum (b) Average (c) Maximum

Figure 5.10: Plots of betweenness centrality for network family **B** with $k \sim Pois(5)$. The UDA and CMA networks have a global clustering coefficient of $C = 0.13$. A trend of increasing average and maximum betweenness centrality is observed between random, UDA and CMA networks, respectively.

and 5.10. However, since UDA and CMA networks share the same degree sequence and global clustering coefficient differences in these metrics between UDA and CMA can only be due to increased degree-degree correlation and negatively skewed distribution of degree-dependent clustering. It has previously been noted that increased assortativity corresponds to an increase in average path length [79] and this will be compounded by the higher-degree nodes (which inevitably serve as central hubs) being more clustered. Similarly, an increase in diameter (a function of path length) will be due to these highly clustered high-degree nodes. Finally, Figures 5.10b and 5.10c show a significant increase in average and maximum betweenness centrality between UDA and CMA networks. This is yet another manifestation of the presence of these highly-clustered high-degree nodes.

Table 5.2 presents a comparison between *measured* and *expected* average subgraph counts for the networks in family **B**. Whereas there is good agreement for UDA networks, it is observed that CMA networks have produced by-products other than what was expected at random, e.g., an additional 50% $G_{\boxslash}$ have appeared as by-products. The effects of finite size have been exacerbated by aggregating clustered subgraphs around higher degree nodes, effectively excluding lower to medium degree nodes during this part of the connection process. Within this densely connected component it is easy to envisage a situation where adding only a single edge may create additional (unwanted) subgraphs. This highlights the fact that whilst the total number of $G_{\triangle}$ is preserved (as evidenced by identical global clustering), the way these subgraphs contribute to higher-order structure can vary significantly.

This Section has highlighted that control over the choice of subgraph families and their distributions makes it possible to flexibly explore the solution space of networks with the same degree distribution and global clustering. This in turn provides us with the means to investigate specific areas of this solution space as well as further my understanding of how network metrics deal with such diversity.

	c4	d4	e4	t3
Random	0	0	79	21
UDA	243	504	587	718
CMA	232	743	772	691
Expected	243	504	619	741

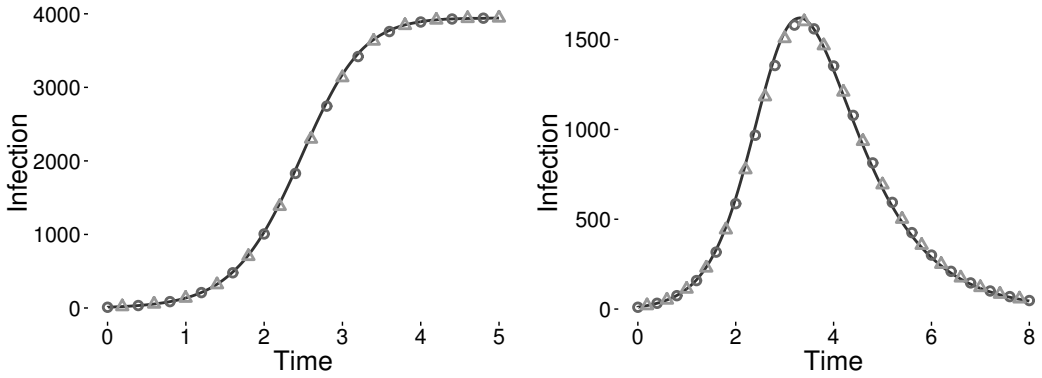
Table 5.2: Subgraph counts for network **B** ($N = 5000$, $k \sim Pois(5)$ and $C = 0.13$). The counts are unique. The expected counts are computed by adding the total counts from the subgraph sequences, dividing them by the subgraphs' node cardinality, and adding these figures to the number of subgraphs found as by-products in the random network. The counts for $t3$ are for $G_{\triangle}$ subgraphs that do not appear in any other subgraphs.

5.3.4 Does higher-order structure matter?

In order to answer this question I make use of the network families **A** and **B** detailed above and test the impact of higher-order structure by considering the outcome and evolution of widely used dynamics on networks, namely, SIS, SIR and the complex contagion model.

For each network type in families **A** and **B** a series of networks were generated. For each network I performed a single Gillespie realisation of the SIS, SIR and complex contagion epidemics. The mean time evolution of infectious prevalence was then calculated, plotted and compared between network types. Complex contagion dynamics was simulated in a similar way but without recovery and remembering that a single infectious contact was usually not sufficient to result in an infected node. Different thresholds of infection and infectious seeds were used and these are specified in figure captions. Matlab code for the SIS and SIR Gillespie algorithms is available from `https://github.com/martinritchie/Dynamics`.

I know by construction that members of network family **A** were generated using different subgraphs and Section 5.3.3 has shown that observable differences were found between networks in terms of average path length, betweenness centrality and subgraph composition. Despite this, Figure 5.11, which show the time evolution for SIS and SIR dynamics respectively, illustrate that these dynamics can display a certain degree



(a) SIS (b) SIR

Figure 5.11: SIS and SIR epidemic dynamics for network family **A**. The random, Big-V and UDA data has been plotted with a solid line, circle and triangle markers respectively. The SIS and SIR epidemics represent the average of single Gillespie simulations on each of the 1000 network realisations from each network generation algorithm. The SIS and SIR epidemics were seeded with an initial infectious seed of $I_0 = 10$ and had a per link rate of infection of $\tau = 1$ and recovered independently at rate $\gamma = 1$.

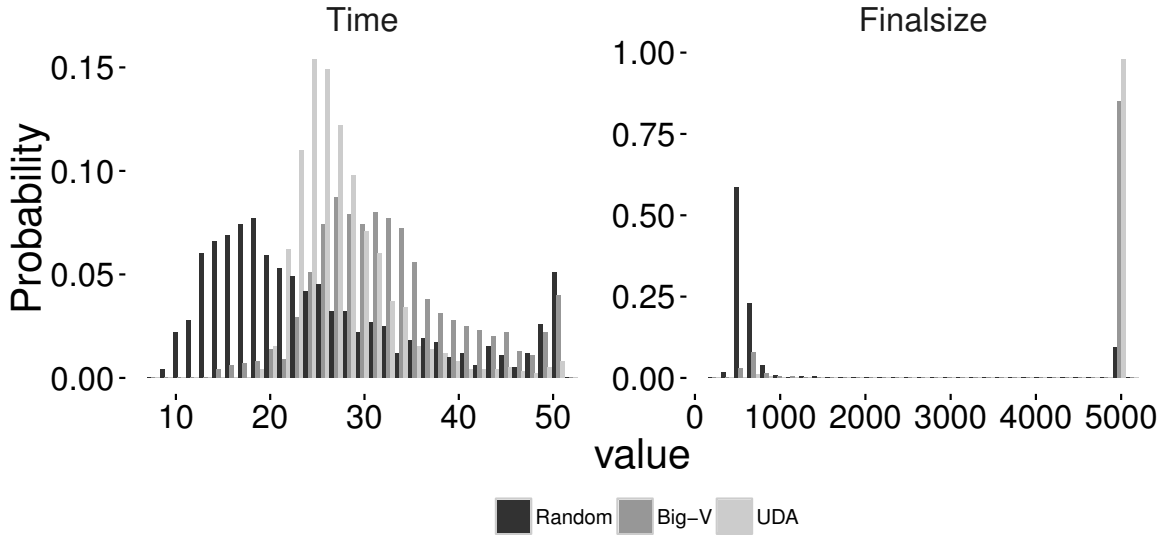
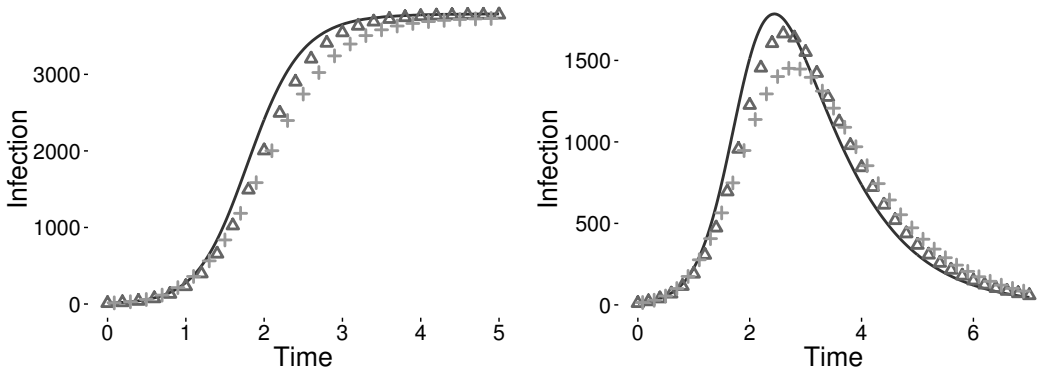


Figure 5.12: Complex contagion dynamics for network family **A**. The complex contagion epidemics I parametrised an initial infectious seed of $I_0 = 250$ and a fixed threshold of infection of $r = 2$.



(a) SIS (b) SIR

Figure 5.13: SIS and SIR epidemic dynamics for network family **B**. The random, UDA and CMA data has been plotted with a solid line, triangle and cross markers respectively. The SIS and SIR epidemics represent the average of single Gillespie simulations on each of the 1000 network realisations from each network generation algorithm. The SIS and SIR epidemics were seeded with an initial infectious seed of $I_0 = 10$ and had a per link rate of infection of $\tau = 1$ and recovered independently at rate $\gamma = 1$.

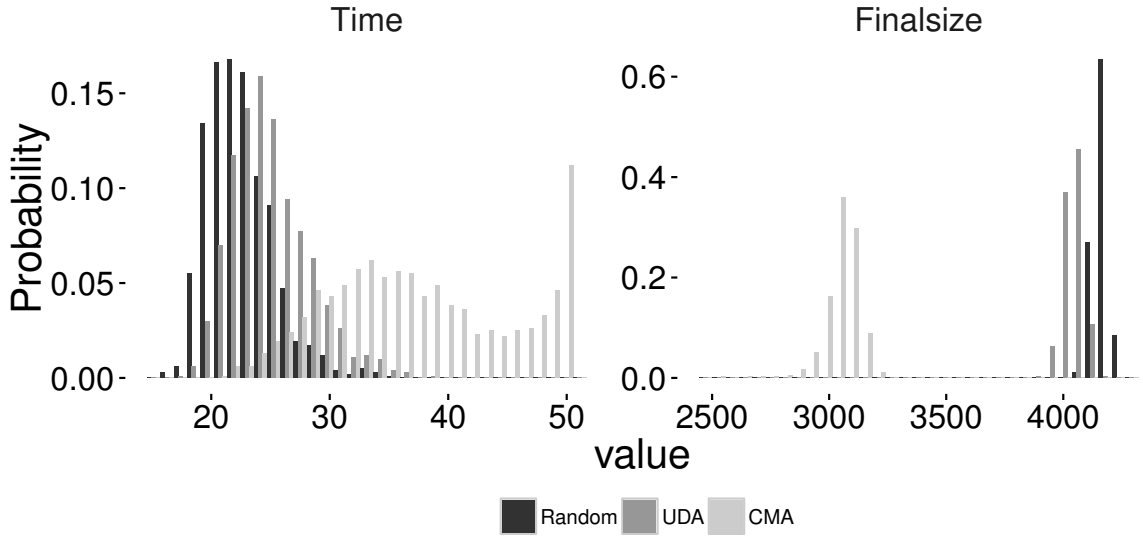


Figure 5.14: Complex contagion dynamics for network family **B**. The complex contagion epidemics had an initial infectious seed of $I_0 = 1000$ and a fixed threshold of infection of $r = 3$.

of insensitivity to these differences in structure. In this case, it is the SIR dynamics that show the greatest difference, in peak infectious prevalence (Figure 5.11b) albeit quite marginal.

In contrast, complex contagion dynamics do show sensitivity to structural differences found between Big-V and UDA networks. Figure 5.12 reveals that for UDA networks the epidemic fully percolates in almost 100% of the simulations instead of only 80% of the cases for Big-V networks and that epidemics on UDA networks achieve this steady state in less time. This indicates that whilst UDA networks operate in the super critical regime, Big-V networks are closer to the transition point. Locating this transition is possible but is beyond the scope of this paper.

When network family **A** is used, the networks' degree distribution and clustering appear to be the main determinants of the time evolution and outcome of the SIS and SIR epidemics. In contrast, when network family **B** is used, Figures 5.13 and 5.14 shows that all dynamics considered are impacted by differences in network topology. For Figures 5.13a and 5.13b, a trend of inhibited spread of infection is observed from the random to UDA to CMA networks. It has already been shown that clustering slows the spread of infection [20, 33], and I see that this effect dominates over higher assortativity, which usually leads to faster initial spread of the epidemic [38]. Similarly, Figure 5.14 which shows the distribution of the final epidemic size for the complex contagion dynamics reveals that: (a) the higher clustering observed in the UDA networks fails to have a significant impact when compared to the random network equivalent and (b) the CMA networks significantly slow the pace of the epidemic as well as reduce its final size compared to both random and UDA networks. Hence, for the UDA and CMA networks where both degree distribution and global clustering are identical the observed differences are explained by the combined effect of varying distributions of subgraph around nodes and varying prevalence of subgraphs (both of which are related to one another to some extent) as shown by Table 5.2.

Taken together, my simulation data shows that even though the proposed algorithms construct networks with identical degree sequence and global clustering, these networks can give rise to measurable differences in resulting epidemics, be it in time evolution or final outcome. With the exception of SIS and SIR epidemics on network family **A**

(still with some small differences) I found significant differences in all other instances. A more systematic investigation of more network models and wider parameter range for the dynamics is needed but left to future work.

5.4 Discussion

In this paper, I have described two novel network generating algorithms that strictly preserve a given degree sequence whilst permitting control over the building blocks of the network and enabling the tuning of global clustering. I have compared these algorithms to one another as well as to the widely used Big-V rewiring algorithm. Using my algorithms I have empirically demonstrated that it is possible to create networks that are identical with respect to degree sequence and global clustering, yet elicit significant differences in network metrics and in the outcome of dynamical processes unfolding on them. I have presented evidence to suggest that the methods sample from different areas of the network state space and that these sampling variations do matter.

Of the two algorithms proposed, UDA is the simplest to use. I believe that this algorithm, when parametrised with complete subgraphs, would be more likely to yield analytical results. Note that whilst varying levels of clustering can be achieved and estimated before network construction it is not possible to target a specific level of clustering, due to the emergent nature of the distribution of subgraphs around nodes. When constructing networks with incomplete subgraphs the UDA must decompose a certain number of hyperstubs back into stubs when generating sequences associated with incomplete subgraphs, which will introduce some bias. However, this source of bias is removed when using only complete subgraphs.

The CMA algorithm is more complex but also more versatile. Being able to build networks based on pre-specified distributions of subgraphs alongside a given degree sequence, and preserve both, is highly novel. However this algorithm also contains a source of bias. In this case, when finding a node to accommodate a certain number of hyperstubs the algorithm only considers nodes of suitably high degree. I conjecture that the algorithm should consider all nodes, regardless of degree, uniformly and if an invalid selection is made the algorithm should restart the process anew. However,

this may result in prohibitive running times, especially when the total number of stubs contained within hyperstubs is close to the total number of stubs specified by the degree sequence.

The proposed connection procedure for subgraph based networks has revealed some surprising results. I found evidence that the number of self, multi-edges and erroneous $G_{\triangle}$ subgraphs in these networks is less than what is found in the equivalent configuration model networks. I should also point out that the connection procedure is not without its source of bias, namely, my reliance on the repetition procedure whereby if a hyperstub selection results in self or multi-edges I return the hyperstubs to their respective bins and make a new selection. The alternative would be the refusal method, whereby an incompatible selection of hyperstubs requires the whole process to start anew. However, such approach leads to prohibitive running times. Note that the bias of the implemented connection process method may be offset by the overall reduction in self and multi-edges when connecting subgraphs. However, all these points ideally warrant supporting analytical results or, atleast, further computational evidence.

The proposed models are unique and although the methods I implemented do suffer from biases, they were critical to being able to generate the desired networks. Importantly, there is currently no way to assess or measure the extent of these biases. This is because for a given degree distribution and given global clustering coefficient, there is currently no ground truth model nor is the entire state-space of such networks known in full. In light of this I have taken the pragmatic step to focus on characterising the network structure in terms of diversity. Being able to obtain diversity within networks sampled from the same part of the state space of such networks will be a critical component of constructing suitable null models.

In this respect, I have shown that significant diversity in networks with identical degree distribution and global clustering can be elicited. This has occurred in two ways: (1) by construction, i.e. changing subgraph families or redistributing the same number of subgraphs, and (2) unexpectedly, through the emergence of by-products. I conjecture that any controlled – or believed to be controlled – network generation algorithm will yield by-products, unless heuristic constraints are introduced to reduce the likelihood of subgraphs sharing lower-order subgraph components for example. As witnessed in my

results, even configuration model networks lead to a large number of loops with 4, 5, and 6 nodes (longer cycles were not measured). This problem can only be exacerbated when control of more sophisticated structures is implemented. As such, care has to be taken when parametrising algorithms. For example, one would need to specify a relatively large number $G_{\hexagon}$ subgraphs in a network's construction to impact the subgraph count beyond what one would observe by chance in a random network. More surprisingly, as I witnessed with $G_{\boxslash}$ subgraphs in the CMA networks from network family **B**, significant numbers of subgraph by-products can appear in addition to what was observed in the random networks depending on how one wishes to place the subgraphs around nodes.

I have seen that by using a very modest selection of subgraphs, I have been able to substantially influence dynamics running on the network, particularly complex contagion dynamics. All results relating to this model indicate that constraining a network by degree sequence and clustering is not sufficient to accurately predict the course of the epidemic. More importantly, the results appear to suggest that the location of the critical regime depends on the higher-order structure of the network (above and beyond clustering).

Being able to generate networks with different structural properties or higher-order structure is a key feature of any network construction algorithm. However, if such structural details do not impact on dynamics unfolding on the network, then models for such dynamics can rely with high confidence on a limited set of network descriptors. Although degree sequence, degree-degree correlations and global clustering coefficient were observed to be the main drivers of disease transmission in models such as SIS and SIR, I found it not to be true in general. This is an important finding because one should remember that the dynamics simulated here are modest in complexity, when compared to models of neuronal dynamics for example, and yet, I were able to elicit significant differences by simply tuning the network structure above and beyond triangles. This implies that determining the role and impact of higher-order structure may yet hold and reveal many important and surprising results.

Acknowledgements: Martin Ritchie gratefully acknowledges EPSRC (Engineering and Physical Sciences Research Council) and the University of Sussex for funding for

his PhD. I would also like to thank Dr J.C. Miller for useful discussions on the complex contagion model [47], and for sharing his code for simulating the complex contagion model on networks [48].

5.5 Appendix

5.5.1 Integer partitions

The set of partitions of a positive integer, k, lists all possible ways of writing k as the sum of other positive integers. For example, the partition space of 3 is: $\{\{3\},\ \{2,1\},\ \{1,1,1\}\}$. The number of ways to partition an integer is given by the *partition function*. For this derivation $p(k)$ is used to denote the partition function evaluated at k, i.e., the number of partitions of the integer k. In the case above, $p(3) = 3$. For $k = 1, 2, 3, 4, 5, 6 \ldots$ the partition function returns $p(k) = 1, 2, 3, 5, 7, 11, \ldots$ respectively and by convention $p(0) = 1$ and $p(-k) = 0$.

This function can be used to calculate the number of times an integer $\alpha < k$ appears in the partitions of k.

First compute the number of partitions in which α will appear atleast once: write k as a partition in the following way $\{k - \alpha, \alpha\}$ and now list all remaining partitions of $k - \alpha$,

$$\begin{aligned}
&\{k - (\alpha + 0), \alpha\}, \\
&\{k - (\alpha + 1), \alpha, 1\}, \\
&\{k - (\alpha + 2), \alpha, 2\},\ \{k - (\alpha + 2), \alpha, 1, 1\}, \\
&\{k - (\alpha + 3), \alpha, 3\},\ \{k - (\alpha + 3), \alpha, 2, 1\},\ \{k - (\alpha + 3), \alpha, 1, 1, 1\}, \\
&\ldots,
\end{aligned} \tag{5.13}$$

This forms a bijection between the partitions of $(k - \alpha)$ and partitions of k where α appears atleast once. Similarly it is possible form a bijection between $(k - m\alpha)$ and partitions of k where α appears *atleast* m times, of which there will be $p(k - m\alpha)$ partitions. Using the cumulative property of this expression it is possible to compute the number of partitions in which α appears *exactly* m times

$$p(k - \alpha(m - 1)) - p(k - \alpha m)$$

multiplying this by m and adding over all multiples of α, $m : m\alpha \leq k$ will give the

number of times that α appears in the partitions of k

$$p(k,\alpha) = \sum_{m=1}^{\lfloor \frac{k}{\alpha} \rfloor} m\big[p(k-\alpha m)) - p(k-\alpha(m+1))\big].$$

5.5.2 Pseudocode for UDA

input : $D = (d_1, d_2, \ldots, d_N)$, $G = \{G_1, G_2, \ldots, G_l\}$

output: $H \in \mathbb{N}_0^{l \times N}$.

Variables

D: degree sequence, N: number of nodes,

G: set of subgraphs, l: number of subgraphs,

g_i: subgraph adjacency matrix, X_k: solution space for degree k,

H: hyperstub degree sequence

Procedure

for *Each subgraph,* G_i **do**

% Identify the degree sequences of the subgraphs.

$s_i = \sum g_i$

% Take the unique elements.

$s_i = unique(s_i)$

end

% Concatenate into a single vector.

$S = (s_1, s_2, \ldots s_l)$

for $k = 1, 2, \ldots k_{max}$ **do**

% $X_k(i, :)$ denotes a hyperstub arrangement for a degree k node.

$X_k = diorecur(S, k)$

end

$\vdots$

5.5.3 Pseudocode for CMA

⋮

for n = *1,2,...,* N **do**

  % Take random element from the solution space.

  $r = rand$; $h_n = X_{D(n)}(r, \cdot)$

end

% Concatenate into a single matrix.

$H = (h_1, h_2, \ldots, h_l)$

return

Algorithm 4: Pseudocode for the underdetermined network generation algorithm (UDA). This pseudocode focuses on the salient points of the UDA, namely, how the algorithm draws solutions from the solution space of an underdetermined Diophantine equation to determine the arrangement of hyperstubs around a particular node. Other steps, such as ensuring the handshake lemma is satisfied for both lines and subgraphs, are detailed in Section 5.2.1 and can be viewed in the source code. The output hyperstub degree sequence H must be used as input for a modified configuration model connection process to realise a network, see Section 5.2.3.

input : $D = (d_1, d_2, \ldots, d_N)$, $G = \{G_1, G_2, \ldots, G_l\}$,
$S = \{S_1, S_2, \ldots, S_l\}$.

output: $H \in \mathbb{N}_0^{|s| \times N}$.

Variables

D: degree sequence, N: number of nodes,

G: set of subgraphs, l: number of subgraphs,

S: subgraph sequence, g_i: subgraph adjacency matrix,

$|s|$: number of unique corners in a subgraph, H: hyperstub degree sequence

Procedure

for *Each subgraph,* G_i **do**

% Identify the degree sequence, s, of the subgraph.

$s_i = \sum g_i$, $s_i = unique(s_i)$, $m = length(s_i)$

% p reflects the proportions of hyperstubs

$p_i = (p_1, p_2, \ldots, p_m)$

for $j = 1, 2, \ldots, N$ **do**

% The subgraph sequence is decomposed into a hyperstub

% sequence using the multinomial distribution, M,

% so that $H_i \in \mathbb{N}_0^{m \times N}$

$H_i(j) = M(S_i(j), p_i)$,

end

% H_i' is a sequence of the true stub count

$H_i' = H_i \cdot s_i$

% Sum so that $H_i' \in \mathbb{N}_0^{1 \times N}$

$H_i'(j) = \sum_{\alpha=1}^{m} H_i'(\alpha, j)$

end

⋮

⋮

while *elements of each* H_i *are non-zero* **do**

% Find the largest subgraph degree,

$h_i(j) = \max\{\max\{H_1'\}, \max\{H_2'\}, \cdots, \max\{H_l'\}\}$

% i.e., the j^{th} element of H_i.

% Find all elements of the degree sequence atleast this large and

% select an element from d' at random

$d' = \{d \in D : d \geq m\}$, $\delta = d'(random)$

% pair $H_i(j)$ to δ and update

% δ's available degree and H_i

$\delta = \delta - H_i(j), \;\; H_i(j) = 0$

end

Algorithm 5: Pseudocode for the cardinality matching algorithm (CMA). Other steps, such as ensuring the handshake lemma is satisfied for both lines and subgraphs, are identical to what is used for the UDA and are detailed in Section 5.2.1 and can be viewed in the Matlab source code. The output hyperstub degree sequence H must be used as input for a modified configuration model connection process to realise a network, see Section 5.2.3.

Chapter 6

Discussion

It is possible to consider certain network measures or metrics, such as density of links, degree distribution, degree correlation, clustering etc, in a hierarchical sense. Starting with the most basic and with increasing sophistication such a list could be: edge density, distribution of edges, assortativity and clustering. These canonical descriptors are indeed powerful but they only capture part of the entire network state space. Therefore, the awareness and development of additional network measures and descriptors, such as higher-order structure, to be used alongside these classical measures is an essential next step in our understanding of complex networks. This thesis has demonstrated that networks with equal degree distribution, assortativity and global clustering may still exhibit diverse structure and function, attributable to differences in higher-order structure.

One of the greatest difficulties I have found in generating networks and network-based models of dynamics with controllable higher-order structure is preserving basic network metrics, particularly the degree distribution. In Chapter. 3 the networks were homogeneous and the family of subgraphs fixed. In Chapter. 4 the generality of the ODE SIR model was juxtaposed against its inability to control all moments of the degree distribution. However, it was possible to constrain the first and second moments in addition to clustering. In the third paper both algorithms succeeded in preserving the degree distribution with the CMA achieving generality in respect to both subgraph families and subgraph distributions, see Chapter. 5. In addition to this none of the

proposed algorithms permitted strict control over assortativity. But due to the configuration model-like random connection procedure all networks, with the exception of one in Chapter. 5, were created with an assortativity coefficient of $r \approx 0$. In all work we found that the global clustering coefficient was relatively simple to control.

Underpinning all of the proposed network generating algorithms is the connection process. This, like for the configuration model, will naturally produce multi-graphs. Consequently steps must be taken to mitigate against this when a simple graph is desired. For the configuration model this may be achieved by deleting self edges and collapsing multi-edges down to a single edge (the deleted configuration model). Alternatively, when selecting pairs of nodes for connection if the resulting pair results in a self or multi-edge then either: (a) make a new selection (the matching algorithm) or (b) disregard the current network and start the process from scratch (the refusal algorithm). These same steps may be taken to mitigate against the formation multi-graphs following the subgraph based configuration model connection procedure. But, as in the standard configuration model these processed each have their own advantages and disadvantages as discussed in Chapter. 2. In our implementation we opted to use the refusal algorithm, to (1) preserve the degree sequence and (2) maintain manageable computational times, but at the cost of biasing how we sample from the state space of networks. The question of bias is important, however, in the absence of a ground truth or a feasible unbiased method of sampling we instead have focused on a more pragmatic characterisation of network structure, diversity.

By constructing networks in a random configuration model-like way using families of subgraphs as building blocks, as opposed to only edges, yields multiple ways to construct networks that share the same degree distribution and global clustering coefficient. These networks have been shown to be different using classical structural metrics/measures such as betweenness centrality, average path length, diameter and the distribution of the local clustering coefficient. This analysis has been complemented by counting subgraphs. These counts showed significant differences, beyond those which were expected due to how the networks were constructed. This is important due to the interdependence between network structure and the way dynamics unfold on networks.

This thesis has predominantly studied the impact of higher-order structure on SIS

and SIR dynamics. The results across all three papers consistently show that different higher-order structure manifests in different behaviour of dynamics on the networks. This strongly aligns with the principle of network function being defined by network structure. How generalisable is this result? There can be no claim that we have fully explored the parameter space for the network generating algorithms and dynamics. For example, relatively small average degrees and subgraph sizes were used throughout. A larger average degree would allow for considerably larger and more complex subgraph families to be used in the network construction. Consequently, it can be hypothesised that the effects observed in this work do not represent the full impact potential of higher-order structure on epidemic dynamics. One possible way of more efficiently exploring such a complex parameter space is marrying the network generating models with the type of genetic algorithms proposed by Overbury and Berthouze in [61].

It has also been previously noted that a networks' motif or subgraph topology defines the function that the network serves [49, 69, 70]. In certain cases the behaviour of dynamics is directly dictated by the networks' subgraph topology [19]. These dynamics are typically more sophisticated than the 'free-fall' epidemics used in this work. On the other hand, the Cardinality Matching Algorithm (CMA), see Chapter.5, represents a more sophisticated network construction algorithm than many of the current cutting-edge approaches. Combining these two approaches has considerable research potential but may require further development of the network generating algorithms to include direct and/or weighted subgraphs.

Chapter. 4 presented the *hyperstub configuration model*, a deterministic ODE network based model of SIR dynamics. This model is highly general in respect to the subgraphs that it can incorporate. However, by specifying distribution of subgraphs with no constraints the resulting degree distribution cannot be controlled for. This can be partially mitigated against by constraining the first and second moments of the degree distribution but constraining higher moments will quickly become intractable. Of all network generation algorithms proposed in Chapter. 5 the Underdetermined Sampling Algorithm (UDA) is the most likely to yield a corresponding system of ODEs that capture network based SIR dynamics. This is evidenced in Chapter. 3 where the proposed network generation algorithm is a specific case of the UDA and where

the PGF of the degree distribution for these networks can be written. The hyperstub configuration model currently can only capture SIR dynamics. This is due to the simplifying assumption, *the test node may not transmit infection back to its original source of infection*, this rules out the inclusion SIS dynamics. Developing a paradigm that does not rely on this assumption is needed to extend the model in this regard.

Whilst the main thrust of this investigation has been computational, analytical results always provide an eloquent and alternative perspective when setting out understand network structure and behaviour of subsequent dynamics. PGFs have already been written for the network generating algorithm in Chapter. 3, and the SIR model from Chapter. 4 is similarly PGF based. Currently, analytical results do exists for networks with a known PGF, see [32, 58], including subgraph counts, size of the giant component and percolation results. Similarly, for PGF based SIR dynamics results exist for final epidemic size, see [76]. Further inroads could be made by applying more of such approaches to the network models developed in the thesis. Moreover, I believe that strong probabilistic and combinatorial arguments could be used to derive more rigorous mathematical results and understand limiting cases of the proposed models better. I would very much enjoy such an approach, and I feel that this could also be a next natural step to strengthen my work further.

Whilst it is clear that the degree distribution, assortativity and global clustering coefficient are key determinants of network structure and function, we have shown that higher-order structure also plays an important role in this regard. Much remains to be revealed regarding the full extent of its role, specifically: (1) a more thorough exploration of the state space of networks with a given degree distribution and global clustering coefficient and (2) using more exotic models of network based dynamics to investigate the function of higher-order structure. I strongly believe that as our understanding of complex systems increases, higher-order network structure will take on an increasingly important role in complexity science.

Bibliography

[1] Route views project. URL `http://routeviews.org/`. Accessed: 12-01-2016.

[2] R. Albert, H. Jeong, and A-L Barabási. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, 2000.

[3] F. Ball and O. D. Lyne. Stochastic multi-type SIR epidemics among a population partitioned into households. *Advances in Applied Probability*, 33(1):99–123, 2001.

[4] F. Ball and D. Sirl. An SIR epidemic model on a population with random network and household structure, and several types of individuals. *Advances in Applied Probability*, 44(1):63–86, 2012.

[5] F. Ball, D. Sirl, and P. Trapman. Analysis of a stochastic SIRe epidemic on a random network incorporating household structure. *Mathematical Biosciences*, 224(2):53–73, 2010.

[6] S. Bansal, S. Khandelwal, and L. A. Meyers. Exploring biological network structure with clustered random networks. *BMC Bioinformatics*, 10(1):1–15, 2009.

[7] A-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[8] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An open source software for exploring and manipulating networks. 2009. URL `http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154`.

[9] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D. U. Hwang. Complex networks: Structure and dynamics. *Physics Reports*, 424(45):175 – 308, 2006.

[10] B. Bollobás. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *European Journal of Combinatorics*, 1(4):311 – 316, 1980.

[11] E. Bullmore and O. Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 10(3):186–198, 2009.

[12] R. Cohen, S. Havlin, and D. Ben-Avraham. Efficient immunization strategies for computer networks and populations. *Physical Review Letters*, 91(24):247901, 2003.

[13] L. Decreusefond, J-S. Dhersin, P. Moyal, and V. C. Tran. Large graph limit for an sir process in random network with heterogeneous connectivity. *The Annals of Applied Probability*, 22(2):541–575, 2012. URL `http://www.jstor.org/stable/41713336`.

[14] K. T. D. Eames. Modelling disease spread through random and regular contacts in clustered populations. *Theoretical Population Biology*, 73(1):104 – 111, 2008.

[15] P. Erdős and A. Rényi. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5:17–61, 1960.

[16] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[17] D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22(4): 403 – 434, 1976.

[18] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[19] L. L. Gollo, C. Mirasso, O. Sporns, and M. Breakspear. Mechanisms of zero-lag synchronization in cortical motifs. *PLoS Computational Biology*, 10(4):1–17, 04 2014.

[20] D. M. Green and I. Z. Kiss. Large-scale properties of clustered networks: implications for disease dynamics. *Journal of Biological Dynamics*, 4(5):431–445, 2010.

[21] J. L. Guillaume and M. Latapy. A realistic model for complex networks. *arXiv preprint cond-mat/0307095*, 2003.

[22] J. A. P. Heesterbeek. *Mathematical epidemiology of infectious diseases: model building, analysis and interpretation*, volume 5. John Wiley & Sons, 2000.

[23] T. House. Generalised network clustering and its dynamical implications. *Advances in Complex Systems*, Vol.13(No.3):281–291, 2010.

[24] T. House. Generalized network clustering and its dynamical implications. *Advances in Complex Systems*, 13(03):281–291, 2010.

[25] T. House and M. J. Keeling. Household structure and infectious disease transmission. *Epidemiology and infection*, 137(05):654–661, 2009.

[26] T. House and M. J. Keeling. The impact of contact tracing in clustered populations. *PLoS Computational Biology*, 6(3):e1000721, 2010.

[27] T. House and M. J. Keeling. Insights from unifying modern approximations to infections on networks. *Journal of the Royal Society Interface*, 8(54):67–73, 2011.

[28] T. House, G. Davies, L. Danon, and M. J. Keeling. A motif-based approach to network epidemics. *Bulletin of Mathematical Biology*, 71(7):1693–1706, 2009.

[29] S. Itzkovitz, R. Milo, N. Kashtan, G. Ziv, and U. Alon. Subgraphs in random networks. *Physical Review E*, 68:026127, 2003.

[30] A. Jamakovic, P. Mahadevan, A. Vahdat, M. Boguná, and D. Krioukov. How small are building blocks of complex networks. *arXiv preprint arXiv:0908.1143*, 2009.

[31] S. Janson, M. Luczak, and P. Windridge. Law of large numbers for the sir epidemic on a random graph with given degrees. *Random Structures & Algorithms*, 45(4): 726–763, 2014.

[32] B. Karrer and M. E. J. Newman. Random graphs containing arbitrary distributions of subgraphs. *Physical Review E*, 82:066118, 2010.

[33] M. J. Keeling. The effects of local spatial structure on epidemiological invasions. *Proceedings of the Royal Society of London B: Biological Sciences*, 266(1421):859–867, 1999.

[34] W. O. Kermack and A. G. McKendrick. A contribution to the mathematical theory of epidemics. 115(772):700–721, 1927.

[35] B. J. Kim. Performance of networks of artificial neurons: The role of clustering. *Physical Review E*, 69(4):045101, 2004.

[36] I. Z. Kiss and D. M. Green. Comment on "properties of highly clustered networks". *Physical Review E*, 78:048101, 2008.

[37] I. Z. Kiss and P. L. Simon. New moment closures based on a priori distributions with applications to epidemic dynamics. *Bulletin of mathematical biology*, 74(7): 1501–1515, 2012.

[38] I. Z. Kiss, D. M. Green, and R. R. Kao. The effect of network mixing patterns on epidemic dynamics and the efficacy of disease contact tracing. *Journal of The Royal Society Interface*, 5(24):791–799, 2008.

[39] H. Klein-Hennig and A. K. Hartmann. Bias in generation of random graphs. *Physical Review E*, 85:026101, 2012.

[40] V. Latora and M. Marchiori. Efficient behavior of small-world networks. *Physical Review Letters*, 87:198701, Oct 2001.

[41] J. Lindquist, J. Ma, P. Driessche, and F. H. Willeboordse. Effective degree network disease models. *Journal of Mathematical Biology*, 62(2):143–164, 2010.

[42] C. Z. Lo, T. Su, C. Huang, C. Hung, W. Chen, T. Lan, C. Lin, and E. T. Bullmore. Randomization and resilience of brain functional networks as systems-level

endophenotypes of schizophrenia. *Proceedings of the National Academy of Sciences*, 112(29):9123–9128, 2015.

[43] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat. Systematic topology analysis and generation using degree correlations. *SIGCOMM Computer Communication Review*, 36(4):135–146, 2006.

[44] V. Marceau, P. Noël, L. Hébert-Dufresne, A. Allard, and L. J. Dubé. Adaptive networks: Coevolution of disease and topology. *Physical Review E*, 82:036116, 2010.

[45] J. C. Miller. Percolation and epidemics in random clustered networks. *Physical Review E*, 80:020901, Aug 2009.

[46] J. C. Miller. A note on a paper by Erik Volz: Sir dynamics in random networks. *Journal of mathematical biology*, 62(3):349–358, 2011.

[47] J. C. Miller. Complex contagions and hybrid phase transitions. *Journal of Complex Networks*, page cnv021, 2015.

[48] J. C. Miller. Personal communication. 2015.

[49] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594): 824–827, 2002. ISSN 0036-8075.

[50] R. Milo, N. Kashtan, S. Itzkovitz, M. E. J. Newman, and U. Alon. On the uniform generation of random graphs with prescribed degree sequences. *arXiv preprint cond-mat/0312028*, 2003.

[51] Y. Moreno, R. Pastor-Satorras, and A. Vespignani. Epidemic outbreaks in complex heterogeneous networks. *The European Physical Journal B - Condensed Matter and Complex Systems*, 26(4):521–529, 2002.

[52] Y. Moreno, J. B. Gómez, and A. F. Pacheco. Epidemic incidence in correlated complex networks. *Physical Review E*, 68(3):035103, 2003.

[53] M. E. J. Newman. Assortative mixing in networks. *Physical Review Letters*, 89: 208701, 2002.

[54] M. E. J. Newman. Spread of epidemic disease on networks. *Physical Review E*, 66: 016128, 2002.

[55] M. E. J. Newman. Properties of highly clustered networks. *Physical Review E*, 68: 026121, 2003.

[56] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[57] M. E. J. Newman. *Networks: an introduction*. Oxford University Press, 2009.

[58] M. E. J. Newman. Random graphs with clustering. *Physical Review Letters*, 103: 058701, 2009.

[59] V. Nicosia, M. Valencia, M. Chavez, A. Díaz-Guilera, and V. Latora. Remote synchronization reveals network symmetries and functional modules. *Phys. Rev. Lett.*, 110:174102, Apr 2013. doi: 10.1103/PhysRevLett.110.174102. URL `http://link.aps.org/doi/10.1103/PhysRevLett.110.174102`.

[60] D. J. P. O'sullivan, G. J. O'Keeffe, P. G. Fennell, and J. P. Gleeson. Mathematical modelling of complex contagion on clustered networks. *Frontiers in Physics*, 3:71, 2015.

[61] P. Overbury and L. Berthouze. Using novelty-biased GA to sample diversity in graphs satisfying constraints. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 1445–1446. ACM, 2015.

[62] N. Pržulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23(2):e177–e183, 2007.

[63] N. Pržulj, D. G. Corneil, and I. Jurisica. Modeling interactome: scale-free or geometric? *Bioinformatics*, 20(18):3508–3515, 2004.

[64] J. M. Read and M. J. Keeling. Disease evolution on networks: the role of contact structure. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 270(1516):699–708, 2003.

[65] M. Ritchie, L. Berthouze, T. House, and I. Z. Kiss. Higher-order structure and epidemic dynamics in clustered networks. *Journal of Theoretical Biology*, 348:21 – 32, 2014.

[66] M. Ritchie, L. Berthouze, and I. Z Kiss. Beyond clustering: mean-field dynamics on networks with arbitrary subgraph composition. *Journal of Mathematical Biology*, 72(1-2):255–281, 2016.

[67] M. Ritchie, L. Berthouze, and I. Z Kiss. Generation and analysis of networks with a prescribed degree sequence and subgraph family: Higher-order structure matters. *Journal of complex networks (in press)*, 2016.

[68] M. Á. Serrano and M. Boguñá. Tuning clustering in random networks with arbitrary degree distributions. *Physical Review E*, 72:036133, Sep 2005.

[69] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of escherichia coli. *Nature genetics*, 31(1):64–68, 2002.

[70] O. Sporns and R. Ktter. Motifs in brain networks. *PLoS Biology*, 2(11), 10 2004.

[71] O. Sporns, G. Tononi, and R. Kötter. The human connectome: a structural description of the human brain. *PLoS Computational Biology*, 1(4):e42, 2005.

[72] S. H. Strogatz. Exploring complex networks. *Nature*, 410(6825):268–276, 2001.

[73] M. Taylor, P. L. Simon, D. M. Green, T. House, and I. Z. Kiss. From markovian to pairwise epidemic models and the performance of moment closure approximations. *Journal of Mathematical Biology*, 64(6):1021–1042, 2011.

[74] E. Volz. Random networks with tunable degree distribution and clustering. *Physical Review E*, 70:056115, Nov 2004.

[75] E. M. Volz. SIR dynamics in random networks with heterogeneous connectivity. *Journal of Mathematical Biology*, 56(3):293–310, 2007.

[76] E. M. Volz, J. C. Miller, A. Galvani, and L. A. Meyers. Effects of heterogeneous and clustered contact patterns on infectious disease dynamics. *PLoS Computational Biology*, 7(6):1–13, 06 2011.

[77] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[78] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *nature*, 393(6684):440–442, 1998.

[79] R. Xulvi-Brunet and I. M. Sokolov. Reshuffling scale-free networks: From random to assortative. *Physical Review E*, 70(6):066102, 2004.